# BRB-ArrayTools
# Version 4.4

# User's Manual

by

Dr. Richard Simon
Biometrics Research Branch
National Cancer Institute

and

BRB-ArrayTools Development Team
The EMMES Corporation

February, 2014

# Table of Contents


**Table of Contents** 2

**Introduction** 6
- **Purpose of this software** 6
- **Overview of the software's capabilities** 6
- **A note about single-channel experiments** 10

**Installation** 12
- **System Requirements** 12
- **Installing the software components** 12
- **Loading the add-in into Excel** 13

**Collating the data** 15
- **Overview of the collating step** 15
- **Input to the collating step** 17
  - Input data elements 17
    - Expression data 17
    - Gene identifiers 18
    - Experiment descriptors 19
    - Minimal required data elements 19
    - Required file formats and folder structures 20
- **Using the collation dialogs** 21
  - Collating data using the data import wizard 21
  - Special data formats 27
  - Collating Affymetrix data from CHP files exported into text format 27
  - Importing Affymetrix data from text or binary CEL files 30
  - Importing Affymetrix Gene ST1.0 .CEL files 31
  - Importing RNA-Seq data outputted from Galaxy web tool 31
  - Collating data from an NCI mAdb archive 32
  - Collating GenePix data 33
  - Collating Agilent data 33
  - Collating Illumina expression data 34
  - Collating Illumina methylation data 35
  - Collating from NCBI GEO Import Tool 36
- **Output of the collating step** 36
  - Organization of the project folder 36
  - The collated project workbook 37

**Filtering the data** 40
- **Spot filters** 40
  - Intensity filter 40
  - Spot flag filter 41
  - Spot size filter 41
  - Detection call filter 41

**Transformations** **41**
Normalization 42
Median normalization 42
Housekeeping gene normalization 42
Lowess normalization 42
Print-tip Group/Sub Grid normalization 43
Single channel data normalization 43
Quantile normalization 43
Normalization by specified target intensity and percentile 44
Normalization by reference array 44
Normalization by array groups 45
Truncation 45
**Gene filters** **45**
Minimum fold-change filter 46
Log expression variation filter 46
Percent missing filter 46
Percent absent filter 46
Minimum Intensity filter 47
**Gene subsets** **47**
Selecting a genelist to use or to exclude 47
Specifying gene labels to exclude 47
Reducing multiple probes/probe sets to one, per gene symbol 47

**Annotating the data** **48**
Defining annotations using genelists 48
User-defined genelists 48
CGAP curated genelists 50
Defined pathways 50
Automatically importing gene annotations 50
Importing gene identifiers for custom annotations 52
Importing annotations from an existing project with the identical chip type 52
Gene ontology 52

**Analyzing the data** **54**
**Scatterplot tools** **54**
Scatterplot of single experiment versus experiment and phenotype averages 54
Scatterplot of phenotype averages 55
**Hierarchical cluster analysis tools** **56**
Distance metric 56
Linkage 57
Cluster analysis of genes (and samples) 59
Cluster analysis of samples alone 60
Interface to Cluster 3.0 and TreeView 61
**Visualizationof samples** **61**
**Using the classification tools** **63**
**Class comparison analyses** **64**
Class comparison between groups of arrays 65

Class comparison between red and green channels 68
Gene Set Comparison Tool 68
Significance Analysis of Microarrays (SAM) 73
**Class prediction analyses 74**
Class prediction 74
Gene selection for inclusion in the predictors 75
Compound covariate predictor 77
Diagonal linear discriminant analysis 77
Nearest neighbor predictor 77
Nearest centroid predictor 78
Support vector machine predictor 78
Cross-validation and permutation p-value 79
Prediction for new samples 82
Binary tree prediction 82
Prediction analysis for microarrays (PAM) 83
**Survival analysis 84**
**Quantitative traits analysis 87**
**Some options available in classification, survival, and quantitative traits tools 87**
Random Variance Model 87
Multivariate Permutation Tests for Controlling Number and Proportion of False Discoveries 88
Specifying replicate experiments and paired samples 90
Gene Ontology observed v. expected analysis 92

**Programmable Plug-In Faciltiy 93**
**Pre-installed plugins 94**
Analysis of variance 94
Random forest 94
Top scoring pair class prediction 94
Sample Size Plug-in 95
Nonnegative matrix factorization for unsupervised sample clustering 95

**Further help 96**
**Some useful tips 96**
Utilities 96
Preference Parameters 96
Download packages from CRAN and BioConductor 97
Excluding experiments from an analysis 97
Extracting genelists from HTML output 98
Creating user-defined genelists 98
Affymetrix Quality Control for CEL files: 99
Using the PowerPoint slide to re-play the three-dimensional rotating scatterplot 100
Stopping a computation after it has started running 101
Automation error 101
Excel is waiting for another OLE application to finish running 102
Collating data using old collation dialogs 103
Example 1 - Experiments are horizontally aligned in one file 103

Example 2 - Experiments are in separate files 108
**Reporting bugs** **111**

**References** **113**

**Acknowledgements** **114**

**License** **114**

# Introduction

## Purpose of this software

BRB-ArrayTools is an integrated software package for the analysis of DNA microarray data. It was developed by the Biometric Research Branch of the Division of Cancer Treatment & Diagnosis of the National Cancer Institute under the direction of Dr. Richard Simon. BRB-ArrayTools contains utilities for processing expression data from multiple experiments, visualization of data, multidimensional scaling, clustering of genes and samples, and classification and prediction of samples. BRB-ArrayTools features drill-down linkage to NCBI databases using clone, GenBank, or UniGene identifiers, and drill-down linkage to the NetAffx database using Probeset ids. BRB-ArrayTools can be used to analyze both single-channel and dual-channel experiments. The package is very portable and is not restricted to use with any particular array platform, scanners, image analysis software or database. The package is implemented as an Excel add-in so that it has an interface that is familiar to biologists. The computations are performed by sophisticated and powerful analytics external to Excel but invisible to the user. The software was developed by statisticians experienced in the analysis of microarray data and involved in research on improved analysis tools. BRB-ArrayTools serves as a tool for instructing users on effective and valid methods for the analysis of their data. The existing suite of tools will be updated as new methods of analyses are being developed.

## Overview of the software's capabilities

BRB-ArrayTools can be used for performing the following analysis tasks:

- **Importing data:** Importing your data to the program and aligning genes from different experiments. The software can load an unlimited number of genes. The previous limitation of 249 experiments has been removed beginning with version 3.4, so that there is no pre-set limitation on the number of experiments. However, memory limitations may apply, which depend on the user's system resources. The entire set of genes may be spotted or printed onto a single array, or the set of genes may be spotted or printed over a "multi-chip" set of up to five arrays. Users may elect whether or not to average over genes which have been multiply spotted or printed onto the same array. Both dual-channel and single-channel (such as Affymetrix) microarrays can be analyzed. A data import wizard prompts the user for specifications of the data, or special interface may be used for Affymetrix or NCI format data. Data should be in tab-delimited text format. Data which is in Excel workbook format can also be used, but will automatically be converted by BRB-ArrayTools into tab-delimited text format.

- **Gene annotations:** Data can be automatically annotated using standard gene identifiers, either using the SOURCE database, or by importing automatic

annotations for specific Affymetrix or Illumina chips using Bioconductor packages. Starting from v4.3, annotation can also be imported from an annotated existing project with identical chip type. If data has been annotated using the gene annotation tool, then annotations will appear with all output results, and Gene Ontology (GO) classification terms may be analyzed for the class comparison, class prediction, survival, and quantitative traits analyses. Gene Ontology structure files may also be automatically updated from the GO website.

- **Filtering, normalization, and gene subsetting:** Filter individual spots (or probesets) based on channel intensities (either by excluding the spot or thresholding the intensity), and by spot flag and spot size values. Affymetrix data can also be filtered based on the Detection Call. For dual-channel experiments, arrays can be normalized by median-centering the log-ratios in each array, by subtracting out a lowess-smoother based on the average of the red and green log-intensities, or by defining a list of housekeeping genes for which the median log-ratio will be zero. For single-channel experiments, arrays can be normalized to a reference array, so that the difference in log-intensities between the array and reference array has median of zero over all the genes on the array, or only over a set of housekeeping genes. The reference array may be chosen by the user, or automatically chosen as the median array (the array whose median log-intensity value is the median over all median log-intensity values for the complete set of arrays). Each array in a multi-chip set is normalized separately. Outlying expression levels may be truncated. Genes may be filtered based on the percentage of expression values that are at least a specified fold-difference from the median expression over all the arrays, by the variance of log-expression values across arrays, by the percentage of missing values, and by the percentage of "Absent" detection calls over all the arrays (for Affymetrix data only). Genes may be excluded from analyses based on strings contained in gene identifiers (for example, excluding genes with "Empty" contained in the Description field). Genes may also be included or excluded from analyses based on membership within defined genelists.

- **Scatterplot of experiment v. experiment:** For dual-channel data, create clickable scatterplots using the log-red, log-green, average log-intensity of the red and green channels, or log-ratio, for any pair of experiments (or for the same experiment). For "M-A plots" (i.e., the plot of log-ratios versus the average red and green log-intensities), a trendline is also plotted. For single-channel data, create clickable scatterplots using the log-intensity for any pair of experiments. All genes or a defined subset of genes may be plotted. Hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases.

- **Scatterplot of phenotype classes:** Create clickable scatterplots of average log-expression within phenotype classes, for all genes or a defined subset of genes. If more than two class labels are present, then a scatterplot is created for each pair of class labels. Hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases.

- **Hierarchical cluster analysis of genes:** Create cluster dendrogram and color image plot of all genes. For each cluster, provides a hyperlinked list of genes, and a lineplot of median expression levels within the cluster versus experiments. The experiments may be clustered separately with regard to each gene cluster. Each gene cluster can be saved and used in later analyses. A color image plot of median expression levels for each gene cluster versus experiments is also provided. The cluster analysis may be based on all data or on a user-specified subset of genes and experiments.

- **Hierarchical cluster analysis of experiments:** Produces cluster dendrogram, and statistically-based cluster-specific reproducibility measures for a given cut of the cluster dendrogram. The cluster analysis may be based on all data or on a user-specified subset of genes and experiments.

- **Interface for Cluster 3.0 and TreeView:** Clustering and other analyses can now be performed using the Cluster 3.0 and TreeView software, which was originally produced by the Stanford group. This feature is only available for academic, government and other non-profit users.

- **Multidimensional scaling of samples:** Produces clickable 3-D rotating scatterplot where each point represents an experiment, and the distance between points is proportional to the dissimilarity of expression profiles represented by those points. If the user has PowerPoint installed, then a PowerPoint slide is also created which contains the clickable 3-D scatterplot. The PowerPoint slide can be ported to another computer, but must be run on a computer which also has BRB-ArrayTools v3.0 or later installed, in order for the clickable 3-D scatterplot to execute.

- **Global test of clustering:** Statistical significance tests for presence of any clustering among a set of experiments, using either the correlation or Euclidean distance metric. This analysis is given as an option under the multidimensional scaling tool.

- **Class comparison between groups of arrays:** Uses univariate parametric and non-parametric tests to find genes that are differentially expressed between two or more phenotype classes. This tool is designed to analyze either single-channel data or a dual-channel reference design data. The class comparison analysis may also be performed on paired samples. The output contains a listing of genes that were significant and hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases. The parametric tests are either t/F tests, or random variance t/F tests. The latter provide improved estimates of gene-specific variances without assuming that all genes have the same variance. The criteria for inclusion of a gene in the gene list is either a p-value less than a specified threshold value, or specified limits on the number of false discoveries or proportion of false discoveries. The latter are controlled by use of multivariate

permutation tests. The tool also includes an option to analyze randomized block design experiments, i.e., take into account influence of one additional covariate (such as gender) while analyzing differences between classes.

- **Class prediction:** Constructs predictors for classifying experiments into phenotype classes based on expression levels. Six methods of prediction are used: compound covariate predictor, diagonal linear discriminant analysis, k-nearest neighbor (using k=1 and 3), nearest centroid, and support vector machines. The compound covariate predictor and support vector machines are only implemented for the case when the phenotype variable contains only two class labels, whereas the diagonal linear discriminant analysis, k-nearest neighbor and nearest centroid may be used even when the phenotype variable contains more than two class labels. Determines cross-validated misclassification rate and performs a permutation test to determine if the cross-validated misclassification rate is lower than would be expected by chance. The class prediction analysis may also be performed on paired samples. The criterion for inclusion of a gene in the predictor is a p-value less than a specified threshold value. For the two-classes prediction problem, a specified limit on the univariate misclassification rate can be used instead of the parametric p-value. In addition, a specified limit on the fold-ratio of geometric means of gene expressions between two classes can be imposed. The output contains the result of the permutation test on the cross-validated misclassification rate, and a listing of genes that comprise the predictor, with parametric p-values for each gene and the CV-support percent (percent of times when the gene was used in the predictor for a leave-one-out cross-validation procedure). The hyperlinks to NCI feature reports, GenBank, NetAffx, or other genomic databases are also included. Permits application of predictive models developed for one set of samples to expression profiles of a separate test set of samples.

- **Binary tree prediction:** The multistage algorithm constructs a binary tree for classifying experiments into phenotype classes based on expression levels. Each node of the tree provides a classifier for distinguishing two groups of classes. The structure of the tree is optimized to minimize the cross-validated misclassification rate. The binary tree prediction method can be based on any of the six prediction methods (compound covariate predictor, diagonal linear discriminant analysis, k-nearest neighbor using k=1 or 3, nearest centroid, and support vector machines). Unlike the class prediction tool, the compound covariate predictor and support vector machines can be used even for the case when the phenotype variable contains more than two class labels. All the other options of this tool are identical to the class prediction tool. The output contains the description of the binary tree and the result of the permutation test on the cross-validated misclassification rate (if requested by the user). For each node of the tree, the result of the permutation test on the cross-validated misclassification rate, and a listing of genes that comprise the predictor are shown. Listings of genes include parametric p-values,

CV-support percent, the hyperlinks to NCI feature reports, GenBank, NetAffx, or other genomic databases.

- **Survival analysis:** Uses Cox regression (with Efron handling of ties) to identify genes that are significantly correlated with survival. The output contains a listing of genes that were significant and hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases. The criteria for inclusion of a gene in the gene list is either a p-value less than a specified threshold value, or specified limits on the number of false discoveries or proportion of false discoveries. The latter are controlled by use of multivariate permutation tests.

- **Quantitative traits analysis:** Correlates gene expression with any quantitative trait of the samples. Either Spearman or Pearson correlation tests are used. The output contains a listing of genes that were significant and hyperlinks to NCI feature reports, GenBank, NetAffx, and other genomic databases. The criteria for inclusion of a gene in the gene list is either a p-value less than a specified threshold value, or specified limits on the number of false discoveries or proportion of false discoveries. The latter are controlled by use of multivariate permutation tests.

- **Gene Ontology comparison tool:** Classes are compared by GO category rather than with regard to individual genes. Provides a list of GO categories that have more genes differentially expressed among the classes than expected by chance. P-values of two permutation tests, LS and KS, are used to select these GO categories. A GO category is selected if the corresponding LS or KS permutation p-value is below the threshold specified by the user. The GO categories are ordered by the p-value of the LS test (smallest first).

- **Gene List comparison tool:** Investigates user-defined genelists and selects a set of genelists with more genes differentially expressed among the classes than expected by chance. P-values of two permutation tests, LS and KS, are used to select these gene lists. A genelist is selected if the corresponding LS or KS permutation p-value is below the threshold specified by the user. The gene lists are ordered by the p-value of the LS test (smallest first).

- **Plugins:** Allows users to share their own analysis tools with other users. Advanced users may create their own analysis tools using the R language, which can then be distributed to other users who have no knowledge of R. Details about the Plugin utility are covered in a separate manual.

## A note about single-channel experiments

All of the tools within BRB-ArrayTools can be equally run on single-channel and dual-channel experiments. For Affymetrix data, it is suggested that the "signal" field produced in MAS 5.0 should be used as the intensity signal. If the "average difference" field is

used as the intensity signal, then genes with negative "average difference" will be automatically thresholded to a value of 1 (log-transformed value of 0), unless the user specifically elects to set those negative “average difference” values to missing during the log-transformation. For sake of convenience of exposition, we will assume dual-channel data throughout this document. We will refer to log-ratios, though a comparable analysis can be run on the log-intensities for single-channel data. We will also refer to "spots" but for Affymetrix arrays the analog of "spot" is the probe set used to detect the expression of a specified gene.

All analyses on log-intensities work exactly the same way as the analyses on log-ratios, but for two exceptions:

1) Data normalization: Dual-channel data is normalized within each array, whereas single-channel data is normalized relative to a designated reference array. See the section on Normalization for more details.

2) Gene filtering: BRB-ArrayTools includes a special filter for single-channel data, to allow users to filter out genes with more than a minimum percentage of “Absent” Affymetrix detection calls.

3) Class prediction: To reduce the dominating effect that genes with overall high intensity might have, the single-channel log-intensities are median-centered on a gene-by-gene basis for all class prediction methods.

# Installation

## System Requirements

BRB-ArrayTools is Windows-compatible software, Windows 98/2000/NT/XP /Vista and Windows 7. It is designed to be used as an add-in to Excel 2000 or later. BRB-ArrayTools v4.2 is compatible on an Apple MacBook pro machine with Windows XP professional installed with Apple's bootcamp and Parallels Desktop for Mac software. The software can also be installed on a 64-bit machine with Windows OS and 64-bit Excel. BRB-ArrayTools is no longer supported for Excel 97.

It is recommended that the user have at least 256MB of RAM. It is possible to run BRB-ArrayTools with less than 256MB of RAM; however, some procedures may be very slow or may not be able to run at all, depending on the size of the dataset.

For producing high-resolution graphics, it is recommended that the user set the display color palette to **True Color**. To change the display color palette settings, click on the following sequence of items: **Windows Start Button → Settings → Control Panel → Display → Settings**.

## Installing the software components

The BRB-ArrayTools software and associated files may be downloaded from the BRB website:

http://linus.nci.nih.gov/BRB-ArrayTools.html

If you have Excel open, please close Excel before installing BRB-ArrayTools.

There are three installation steps:

1. If you do not already have the **Java Runtime Environment, version 1.4.1 or later** on your computer, then you should download and execute the installation file, which installs the jre-7u9-windows-i586.exe onto your computer.

2. If you do not already have the **R 2.15.2** on your computer, then you should download and execute the R-2.15.2-win.exe installation file, which installs the R 2.15.2 software onto your computer. Existing BRB-ArrayTools users need to upgrade their R to 2.15.2.

3. Now you are ready to install **BRB-ArrayTools**. Download and execute the `ArrayTools_v4_3_beta_2.exe` installation file. If you do not already have R 2.15.2 installed, then you should go back to step #2 and install R 2.15.2

first before installing BRB-ArrayTools. If you have a previous version of BRB-ArrayTools installed, then your new version of BRB-ArrayTools will overwrite your previous version.

## Loading the add-in into Excel

To begin using BRB-ArrayTools, you must have BRB-ArrayTools loaded into Excel as an add-in. The BRB-ArrayTools installer now pre-loads BRB-ArrayTools into Excel, so users no longer have to load BRB-ArrayTools into Excel as a separate step as they did with previous versions of BRB-ArrayTools. Users may still load and unload BRB-ArrayTools in Excel manually by going to the **Tools** menu within Excel and clicking on **Add-Ins**.

Previous versions of BRB-ArrayTools required users to load the RServer add-in as a separate step. This extra step is no longer required, so users who had a previous version of BRB-ArrayTools should going to the **Tools** menu within Excel, click on **Add-Ins**, and check to make sure that the RServer add-in (if it is still showing in the list of available add-ins) is *deselected*. Leaving the RServer add-in loaded as a separate add-in may cause BRB-ArrayTools to display spurious error messages such as "This workbook is referenced by another workbook and cannot be closed" when loading and unloading the ArrayTools add-in. For this reason, it is best not to load the RServer as a separate add-in, since the ArrayTools add-in will now automatically load the RServer internally.

If you close an Excel session with BRB-ArrayTools still loaded, then your next Excel session will automatically begin by re-loading BRB-ArrayTools. BRB-ArrayTools will not be unloaded from Excel unless you specifically unload it through the **Add-Ins** dialog.

**A note about security settings**

In order for BRB-ArrayTools to function, you should have your Excel Security Level set to Medium, and you must select the 'Trust all installed add-ins and templates' option. Office XP users must also select 'Trust access to Visual Basic Project'. To change your security settings within Excel, go to the **Tools → Macro → Security…** menu item from your Excel menubar.

**A note about Excel add-in**

Once the Excel add-in is loaded, you will see **ArrayTools** showing in the menu bar at the top of the Excel screen (usually to the right of **Help**). If **ArrayTools** is not shown, then go to **Tools → Add-Ins → Browse → C:/Program Files/ArrayTools/Excel** (or the Excel folder

of your ArrayTools installation folder), and select the ArrayTools.xla file. After a few moments, the **ArrayTools** add-in will be added to your list of ones available and the box next to it will be checked. It is recommended that you close Excel at this point to save this add-in setting, and restart Excel to begin processing your data.

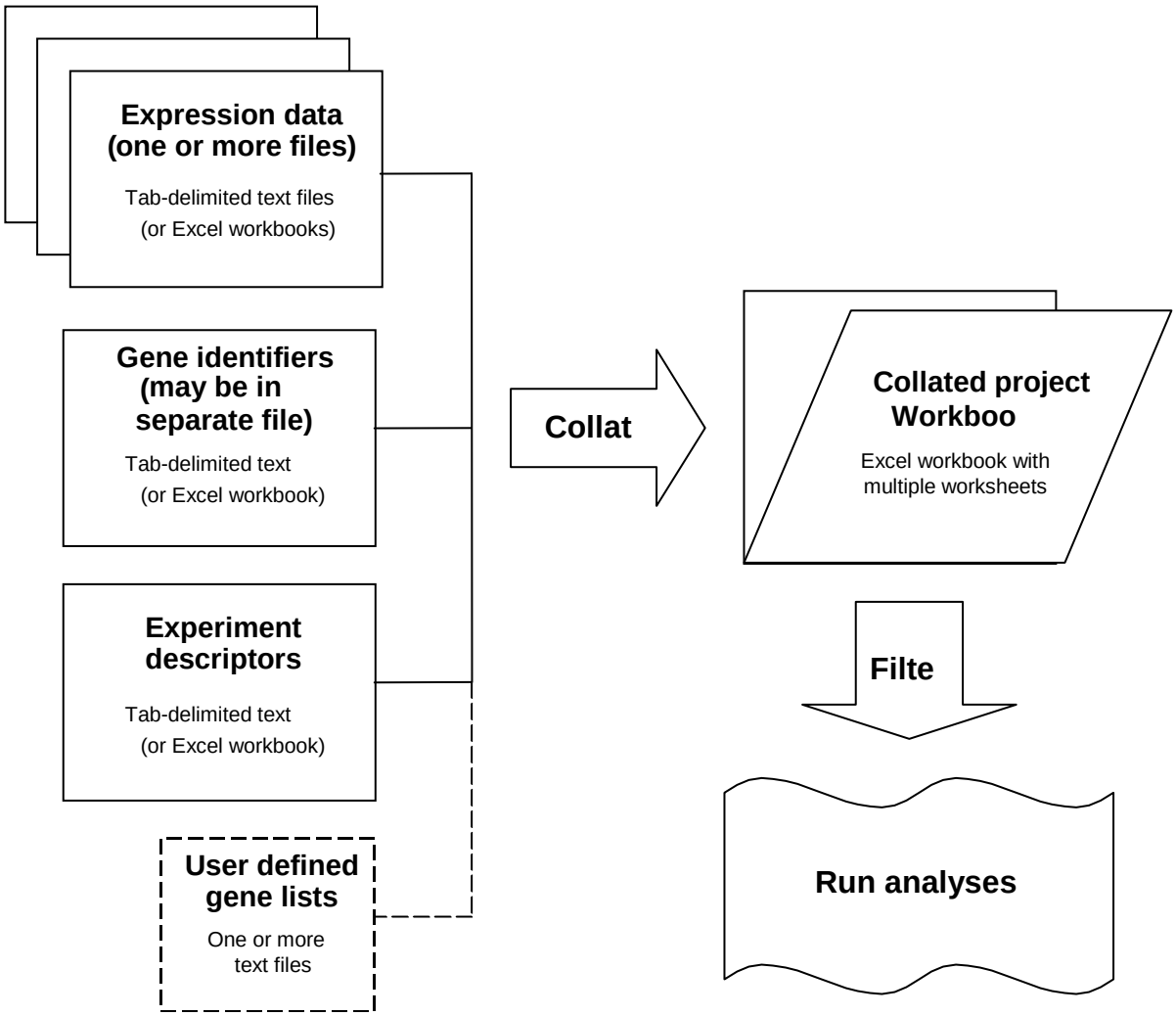
# Collating the data

## Overview of the collating step

**Collating** is a process in which your data are read, sorted, and written to a standard Excel workbook format which is "understood" by BRB-ArrayTools. The user inputs three types of data (expression data, gene identifiers, and experiment descriptors) by entering parameters to describe the file format of each data type. BRB-ArrayTools then processes these data files and produces a **collated project workbook** which is a standard data format used by all the analysis tools included in the software package. Curated gene lists may also be used to define gene functions or pathways, and will be read in at the time of collating. The figure below shows a diagram of the collation process.

**A note about Excel workbooks**

A single Excel file (with an .xls extension) is called a **workbook**, and a workbook may contain one or more spreadsheets called **worksheets**. A tab-delimited ASCII text file with an .xls file extension, though not a true Excel object, is interpreted by Excel to be a workbook containing a single worksheet.


Expression data
(one or more files)
Tab-delimited text files
(or Excel workbooks)
Gene identifiers
(may be in
separate file)
Tab-delimited text
(or Excel workbook)
Experiment
descriptors
Tab-delimited text
(or Excel workbook)
User defined
gene lists
One or more
text files
Collat
Collated project
Workboo
Excel workbook with
multiple worksheets
Filte
Run analyses

# Input to the collating step

The input data files should be tab-delimited ASCII text files. Excel workbooks (where the input data is on the *first* worksheet if there are multiple worksheets within the workbook) may be used instead of tab-delimited text files. However, BRB-ArrayTools will automatically convert those Excel files into tab-delimited text format. If the user wishes to save a copy of the original Excel files, then the user should first save a copy of those original data files into another folder. The user must enter enough information describing the format of these input data files (such as the column designation of each data element) so that BRB-ArrayTools can find and process these data elements from the input files.

## Input data elements

### Expression data

In general, BRB-ArrayTools accepts expression data as either tab-delimited text files, or Excel files. For Affymetrix data, BRB-ArrayTools now also accepts expression data in the CEL file format (see the section entitled Collating Affymetrix data from CEL files).

BRB-ArrayTools accepts two input file formats for the expression data: (1) arrays **horizontally aligned** in one file, or (2) arrays in **separate files**. Both the horizontally aligned and separate files formats can be used with multi-chip sets. If the genes are listed in the same order and there is a one-to-one correspondence between the rows of each of the expression data columns, then the data is said to be **aligned**. The horizontally aligned data format can only be used when the genes are already aligned, while the separate files format may be used whether or not the genes are already aligned. The user has the option whether or not to geometrically average over replicated spots within an array. For multi-chip sets, genes that are multiply-spotted over several arrays can have replicated spots within the same array geometrically averaged, but replicates across different arrays will not be averaged. For multi-chip sets, the individual arrays are normalized separately, but the data for all arrays within a set will be concatenated to form a "virtual array".

The required expression data elements are either the red and green signal intensity columns or the log-ratio column (for dual-channel data), or a single signal intensity column (for single-channel data). If a background subtraction is desired for the dual-channel data, then the red and green background columns are also necessary. In addition, a spot flag or spot size column may optionally be entered, to allow spot filtering using these data elements. For Affymetrix data, the Detection Call (Absent, Marginal, or Present) may be used as the "spot flag" variable for the purpose of flag filtering.

For single channel data, all signal intensity values less than 1 will automatically be thresholded to 1 by default, before the log transformation is applied. Users who do not wish to automatically threshold their data can turn off this default action by selecting the "Do not threshold to 1 (e.g., CodeLink)" checkbox at the time of collating. For

CodeLink data, signal intensity values have already been normalized so that half of all signal intensities on an array are between 0 and 1, so that it does not make sense to threshold these values to 1. When the “Do not threshold to 1” option is selected, then signal intensity values which are negative or 0 will be set to missing, since a log transformation is not valid on such data values. Please note, however, that the “Do not threshold to 1” option is IRREVERSIBLE! Once the negative or 0 values have been set to missing, they can never be thresholded again to 1 by re-filtering the data. Likewise, once the values less than 1 have been thresholded to 1, the negative or 0 values can never be separated from the values in the interval between 0 and 1 and be subsequently set to missing. In order to change the “Do not threshold to 1” option, the data must be re-collated.

## Gene identifiers

Various identifiers can be associated with each spot, such as spot numbers, well numbers, clone names, clone identifiers, probe set identifiers, UniGene identifiers, GenBank accession numbers, etc. The gene identifiers may be located alongside the expression data in the same files, or may be contained in a separate file which is used as a look-up table for the genes in all the arrays. For multi-chip sets using a separate gene identifier file, only one gene identifier file should be used for the entire set of arrays. If the gene identifiers are contained in a separate file, then there must be corresponding columns within the expression data file(s) and the gene identifier file, containing gene ids which can be used for matching the gene identifiers with the expression data.

The column which is designated within BRB-ArrayTools as **clone id** should contain an organization-prefixed clone id (e.g., a prefix such as "IMAGE:", "ATCC:", "TIGR:" etc.). These clone ids can be used to link to clone reports in the NCBI database. Note that clone reports in the NCI mAdb database are only available for clones in the NCI Advanced Technology Center inventory or for other expression array sets that are tracked by BIMAS/CIT/NIH. All clone identifiers found within a **clone id** column that are numeric and have no prefix will be assumed to have a prefix of IMAGE by default.

**Probe set ids** are used to link to feature reports in the NCI mAdb database. Currently feature reports are available only for the Human Genome U133 A and B chips, and for the Mouse Genome U74 A-C chips. For Class Comparison, Class Prediction, and Survival Analysis output, probe set ids are also used for batch query links to NetAffx.

**UniGene cluster ids** and **gene symbols** are used to search for the UniGene annotation mirrored in the NCBI database. **GenBank accession numbers** are used to search for the GenBank annotation which is also mirrored in the NCBI database.

**microRNA Id** can be imported in v4.1 as part of the gene identifiers.

A minimum of one gene identifier is required for use in collating the dataset. However, the user may wish to enter any or all of the above gene identifiers, if they are available, to enhance the usability of the output from the analyses.

## Experiment descriptors

A file containing experiment descriptors must be prepared by the user and input during collation. Each row in the experiment descriptor file represents an experiment in the dataset (except for the first row which is a header row), and each column represents a descriptor variable whose entries are used to describe each of the experiments. For multi-chip sets, each row in the experiment descriptor file should represent an entire set of arrays performed with the same sample, not a separate row for each individual array. The experiment descriptor file should contain *exactly* the same experiments as those to be collated (i.e., it should not contain any extra rows representing experiments which are not represented in the dataset to be collated).

The first column of the experiment descriptor file should contain the names of the experiments. If the expression data is in a separate file for each array, then these names should be the filenames without the ".xls" or ".txt" file extensions. For multi-chip sets with separate data files for each array, these experiment names should be the filenames without the optional "_A", "_B", "_C", "_D", and "_E" suffixes and without the ".xls" or ".txt" file extensions. For horizontally aligned data, the order of the arrays in the experiment descriptor file is assumed to correspond to the order of the arrays in the gene expression data file.

Each succeeding column in the experiment descriptor file contains a descriptor variable which may be used for labeling purposes, for identifying reverse fluor arrays, for classification analyses, for identifying replicate experiments, for matching between paired experiments, or for specifying the plotting order of the experiments when clustering genes. The user can create as many columns of the experiment descriptor file as he/she finds useful for classifying the arrayed samples. There should be no empty columns between the experiment descriptor columns.

## Minimal required data elements

Although there are many optional data elements that may be input into the collating step, the minimal required data elements for all input formats are as follows:

(1) the expression data (either the red and green signals or the log-ratio for dual-channel data, or the single signal intensity for single-channel data),
(2) at least one gene identifier (may be located alongside the expression data or in a separate gene identifier file), and
(3) at least one experiment descriptor column (the experiment id) in the experiment descriptor file.

## Required file formats and folder structures

### Horizontally aligned format

For data in the horizontally aligned format, expression data for all experiments are found in one file. Usually the first few columns will be expected to contain gene identifiers. After the gene identifier columns, there should be a set of columns for each experiment in the dataset, and the *order* of the columns *should be the same* for each experiment. Also, there should be no extraneous columns at the *end* of the file. For example, the file may have columns:

| | | | | | | |
|---|---|---|---|---|---|---|
| Probeset | Signal1 | Detection1 | Signal2 | Detection2 | Signal3 | Detection3 |

However, the following set of columns are illegal for the horizontally aligned data format:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Clone Id | GBAcc | Red_ Exp1 | Green_ Exp1 | Ratio_ Exp1 | Green_ Exp2 | Red_ Exp2 | Ratio_ Exp2 | Filter |

There are two reasons why the above set of columns are illegal. First, the data elements do not appear in the *same order* for experiment 1 (with “Red”, “Green” and “Ratio”) and experiment 2 (with “Green”, “Red” and “Ratio”). Also, there is an extraneous column (“Filter”) at the end of the file which does not belong to any of the experiments. The above file can be corrected by re-ordering the columns as follows:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Clone Id | GBAcc | Filter | Red_ Exp1 | Green_ Exp1 | Ratio_ Exp1 | Red_ Exp2 | Green_ Exp2 | Ratio_ Exp2 |

The “Filter” column will be ignored, since it is neither a gene identifier nor an experiment data element. Also, the “Ratio” columns will be ignored, because the ratio will be automatically computed from the “Red” and “Green” columns.

### Separate files format

For the separate files format, the user should create a data folder consisting *exclusively* of expression data files (one for each array). The column format (i.e., the order of the data elements) should be the *same* in all the expression data files.

# Using the collation dialogs

## Collating data using the data import wizard

To import data into BRB-ArrayTools, the user should go to the

**ArrayTools -> Import data -> Data import wizard**

menu item. A wizard will prompt the user for the data type and location of the data elements described above. The user must specify the following information within the wizard screens:

Data types (e.g., single or dual channel, single array type or multiple array types, etc.)
File type (e.g., horizontally aligned file format or expression data in a separate file for each array)
Expression data
Gene identifiers
Experiement descriptors

### Wizard screen: data types

BRB-ArrayTools can import the following data types using the **Data import wizard**:

| | |
|---|---|
| 1. Dual-channel intensities | Data consists of red and green channel intensities. Background columns may also be included if intensities have not yet been background-subtracted. Data has not yet been log-transformed. |
| 2. Dual-channel ratio or log-ratio | Data consists of ratios (=red/green) or log-ratios (= log2(red/green)). |
| 3. Single-channel intensity or log-intensity | Data from single channel experiments. Data may or may not have already been log-transformed. |
| 4. Affymetrix data | Probeset-level data from Affymetrix GeneChips. Data consists of the Signal (MAS 5.0), Avg Diff (MAS 4.0), or other probeset-level expression summary measure. Data may or may not have already been log-transformed. |

**Important:** Each data column must be clearly labeled with a variable name, and data columns should be tab-delimited. The data file should not contain consecutive tabs, except as necessary to denote missing values in the data columns.

The dual-channel and single-channel data types (data types #1, #2, and #3 above) may also include optional spot flag or spot size columns, which are used to indicate spot quality. Affymetrix data (data type #4 above) may also include detection call or absolute call columns.

Some datasets may fall into more than one category. For instance, the user may have dual-channel data in GenePix files which contain the individual channel intensities as well as a log-ratio column. In that case, the user may elect to import either the individual channel intensities (data type #1 above) or the ratios or log-ratios (data type #2 above), but not both. If the individual channel intensities are imported instead of the ratios or log-ratios, then BRB-ArrayTools will compute the log-ratios from the channel intensities. However, any normalization which had been applied to the log-ratios in the original expression data file will be lost, so the imported data must be re-normalized.

Affymetrix data may be imported as single-channel intensity or log-intensity (data type #3 above) or Affymetrix data (data type #4 above). However, the default filtering criteria are different for these two data types, and the filtered data in the collated project workbook will differ slightly unless the user is careful to use the same filtering criteria in both cases. For Affymetrix data (data type #4 above), the user must specify the chip type (e.g., U133, U74, etc.).

## Wizard screen: file type

BRB-ArrayTools can accept either tab-delimited ASCII text files, or Excel spreadsheets (a single worksheet within an Excel workbook). If the file is an Excel spreadsheet, then BRB-ArrayTools will automatically convert it to tab-delimited text file with the same name.

The expression data can be saved either in a horizontally aligned file or saved in separate files stored in one folder.

### Horizontally aligned file

In this file format, the data columns are organized into array data blocks, where an array data block is a set of consecutive columns containing data from the same array. The data elements must appear in the same order within each array data block. All gene identifier columns must be placed before the first array. If there are any miscellaneous data columns following the last array data block, then the user must either delete those last columns or place them before the first array.

The following figure illustrates an example of a horizontally aligned file with three columns (Green, Red and Flag) in each data block:

Gene Ids Columns | Array 1 Data Block | Array 2 Data Block | Array 3 Data Block

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Unique ID | Well_id | Clone | Description | Green1 | Red1 | Flag1 | Green2 | Red2 | Flag2 | Green3 | Red3 | Flag3 | Green4 |
| 2 | A28102_at | 16027 | IMAGE:809 | caspase 11 | 617 | 676 | 10 | 183 | 238 | 20 | 306 | 294 | 30 | 3 |
| 3 | AB000114_at | 16028 | IMAGE:668 | adhesion select | 5252 | 4011 | 10 | 8657 | 5130 | 20 | 2522 | 1954 | 30 | 31 |
| 4 | AB000115_at | 16029 | IMAGE:767 | adhesion VCAM | 378 | 240 | 0 | 112 | 24 | 0 | 132 | 20 | 0 | 1 |
| 5 | AB000220_at | 4620 | IMAGE:485 | adhesion ELAM | 183 | 154 | 0 | 285 | 153 | 0 | 298 | 182 | 0 | 2 |
| 6 | AB000381_s_at | 4621 | IMAGE:485 | adhesion integri | 840 | 886 | 0 | 847 | 667 | 0 | 471 | 389 | 0 | 4 |
| 7 | AB000409_at | 4622 | IMAGE:486 | adhesion integri | 189 | 173 | 0 | 120 | 253 | 0 | 263 | 276 | 0 | 3 |
| 8 | AB000410_s_at | 4623 | IMAGE:485 | adhesion integri | 602 | 359 | 0 | 305 | 295 | 0 | 409 | 815 | 0 | 4 |
| 9 | AB000449_at | 4624 | IMAGE:485 | adhesion throml | 1167 | 696 | 0 | 284 | 283 | 0 | 528 | 452 | 0 | 6 |
| 10 | AB000450_at | 4625 | IMAGE:485 | adhesion ICAM | 3929 | 3197 | 0 | 2945 | 2716 | 0 | 2809 | 2152 | 0 | 16 |

DualChannelIntensities

Ready

**Important:** If your experiments consist of multiple array types, then each horizontally aligned file must contain arrays of the same type, and you should have one horizontally aligned file for each array type used. The experiments should appear in the *same order* in each horizontally aligned file. Up to five array types (denoted here as 'A', 'B', 'C', 'D' and 'E' for convenience) may be used, and data from these array types will be concatenated to form a "virtual array".

<u>Separate files</u>

In this file format, each array is stored in a separate file. Each file must have identical format, including the same number of header lines before the data lines. The following example illustrates a file with three columns of data:

Data Columns of One Array

| | A | B | C | D |
|---|---|---|---|---|
| 1 | Unique ID | Red | Green | Flag |
| 2 | A28102_at | 617 | 676 | 10 |
| 3 | AB000114_at | 5252 | 4011 | 10 |
| 4 | AB000115_at | 378 | 240 | 0 |
| 5 | AB000220_at | 183 | 154 | 0 |
| 6 | AB000381_s_at | 840 | 886 | 0 |
| 7 | AB000409_at | 189 | 173 | 0 |

Sheet1 / Sheet2 / Sheet3

**Important:** The data folder must contain only expression data files. Extraneous files should be removed before collation. If your experiments consist of multiple array types, do not mix arrays of different types in the same data folder. Data for each array type should be stored in a separate folder, and corresponding samples performed on each array type should have the *same filename* within their respective folders, except that the names may differ by the extension '_A', '_B', etc. (For example, the user may have a folder called 'ChipTypeA' containing files 'PatientID001_A.txt', 'PatientID0234_A.txt' and 'PatID32_A.txt', and another folder called 'ChipB' containing files 'PatientID001_B.txt', 'PatientID0234_B.txt' and 'PatID32_B.txt'.) Up to five array types (denoted here as 'A',

‘B’, ‘C’, ‘D’ and ‘E’ for convenience) may be used, and data from these array types will be concatenated to form a “virtual array”.

**Wizard screen: expression data**

To specify the data columns in the expression data file(s), first select the header line that identifies the data columns as well as the first line of data, and then specify the individual column containing the data elements. For the horizontally aligned file format, the user must also select the columns where data for the first array and second array begin. This helps the Wizard to determine how many columns of data belong to each array, and allows the Wizard to calculate the beginning column for each subsequent array. The columns for each array must be placed together, and there should be no miscellaneous columns inserted between the data columns for each array or after the last column of the last array.

Here is a brief description of the data columns:

Gene identifier column:

- The Unique ID or Well ID or Spot ID column uniquely identifies each spot (or feature) on the array. If this ID appears more than once within the same file, then it is assumed that the clone has been multiply-spotted onto the array. If the data is Affymetrix, then this column is called “Probe Set Name” or ‘Probe Set ID”.

For single channel data:

- The Signal Intensity column contains the signal values of the spots (or features). If the user has Affymetrix data from MAS 4.0, then the “Avg Diff” column may be used as the Signal Intensity column.

For dual channel data:

- The Red and Green Intensity columns indicate the intensity of the red and green signal. Log-ratios of red-to-green will be computed from the red and green background-adjusted values.

- The Red and Green Background columns contains background values to be subtracted from the Red and Green Intensity columns if they are not already background adjusted. These two columns are optional.

Optional data elements common to single and dual channel data:

- The Spot Size column indicates the size of the spot. Some imaging software use the size data for quality control. This column is optional.

- The Spot Flag column indicates the quality of the spot. Some imaging software set the spot flag values such as 0 or 1 to indicates spot quality. A string such as “Failed” or “Pass” may also be used to indicate spot quality. For Affymetrix data, the Detection Call will be used in lieu of the Spot Flag. This column is optional.

**Note:** For single channel data, all signal intensity values less than 1 will automatically be thresholded to 1 by default, before the log transformation is applied. Users who do not wish to automatically threshold their data can turn off this default action by selecting the “Do not threshold to 1 (e.g., CodeLink)” checkbox at the time of collating. For CodeLink data, signal intensity values have already been normalized so that half of all signal intensities on an array are between 0 and 1, so that it does not make sense to threshold these values to 1. When the “Do not threshold to 1” option is selected, then signal intensity values which are negative or 0 will be set to missing, since a log transformation is not valid on such data values. Please note, however, that the “Do not threshold to 1” option is IRREVERSIBLE! Once the negative or 0 values have been set to missing, they can never be thresholded again to 1 by re-filtering the data. Likewise, once the values less than 1 have been thresholded to 1, the negative or 0 values can never be separated from the values in the interval between 0 and 1 and be subsequently set to missing. In order to change the “Do not threshold to 1” option, the data must be re-collated.

## Wizard screen: gene identifiers

Various identifiers such as spot number, well number, clone number, UniGene cluster identifiers, GenBank accession number or gene title can be associated with each spot. They can be either placed alongside the expression data or stored in a separate gene identifiers file. These identifiers will be hyperlinked in the analysis output. For Affymetrix data, the user has the option of downloading the probeset annotation file directly from BRB’s server. These files were originally downloaded from the NetAffx website and have been especially formatted for use with BRB-ArrayTools.

Here is an example of a gene identifiers file:

| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Spot ID | Well_id | Clone | Description | UniGene | GeneBank | Map Location | | |
| 2 | A28102_at | 16027 | IMAGE:80 | IRF-3=interferon regulatory factor | Hs.75254 | IRF3 | 19q13.3-q13.4 | | |
| 3 | AB000114 | 16028 | IMAGE:66 | Receptor protein tyrosine kinase T | Hs.71891 | DDR2 | 1q12-q23 | | |
| 4 | AB000115 | 16029 | IMAGE:76 | HS1= hematopoietic lineage cell s | Hs.14601 | HCLS1 | 3q13 | | |
| 5 | AB000220 | 4620 | IMAGE:48 | delta sleep inducing peptide, imm | Hs.75450 | DSIPI | Xp21.1-q25 | | |
| 6 | AB000381 | 4621 | IMAGE:48 | P-selectin glycoprotein ligand | Hs.79283 | SELPLG | 12q24 | | |
| 7 | AB000409 | 4622 | IMAGE:48 | mrg1=melanocyte-specific nuclea | Hs.82071 | CITED2 | 6q23.3 | | |
| 8 | AB000410 | 4623 | IMAGE:48 | CREG=cellular repressor of E1A- | Hs.5710 | CREG | 1q24 | | |

DualChannelIntensities_GeneIden

Ready

**Important:** If the gene identifiers are in a separate file rather than in the expression data file, then you must specify which gene identifier in the separate gene identifiers file should be used to match against the Spot ID (Well ID, Unique ID or Probe Set ID) of the

expression data file(s). For multi-array sets using a separate gene identifiers file, all gene identifiers should be contained in one file rather than a separate file for each array type.

## Wizard screen: experiment descriptors

In order to facilitate the analysis of your experiments, an experiment descriptors file may be prepared by the user before the collation. If the user does not have experiment descriptors file prepared in advance, the user may elect to have BRB-ArrayTools create a template.

Here is an example of an experiment descriptors file:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Array | Dx | Medulo Type | Medulo Stage | Sex | Age at Dx | Survival (month | SurvStatus | Chemo | AT/RT Site | M Stage | SurvStatusCode | Age Code | Medu |
| 2 | Brain_MD_1 | Medulloblı | Classic | T4M1 | M | 8m | 11 | D | V,C,Cx,VP | | >0 | 1 | 0 | Medu |
| 3 | Brain_MD_2 | Medulloblı | Classic | T2M0 | M | 8yr10m | 5 | D | V,C,Cx,VP | | 0 | 1 | 1 | Medu |
| 4 | Brain_MD_3 | Medulloblı | Classic | T3M0 | M | 6yr | 7 | D | V,C,Cx | | 0 | 1 | 1 | Medu |
| 5 | Brain_MD_4 | Medulloblı | Classic | T3M3 | M | 5yr 3m | 7 | D | V,C,Cx,VP | | >0 | 1 | 1 | Medu |
| 6 | Brain_MD_5 | Medulloblı | Classic | M3 | M | 38yr 2m | 7 | D | V,C | | >0 | 1 | 2 | Medu |
| 7 | Brain_MD_6 | Medulloblı | Classic | T4M0 | F | 7m | 9 | D | V,C,Cx | | 0 | 1 | 0 | Medu |
| 8 | Brain_MD_7 | Medulloblı | Classic | T1M0 | M | 6yr 5m | 14 | D | V,C,Cx | | 0 | 1 | 1 | Medu |

ExpDescrMedulo

Ready

Except for the first row which is a header row, each row represents an experiment in the dataset. The first column should contain the names of the experiments.

**Important:** For data in separate files format, the experiment names should be the name of the file minus the ".xls" or ".txt" file extensions. For data stored in a single file, the experiment names should correspond to the order of the arrays listed in the expression data file.

**Important:** The experiment descriptors file should contain *exactly* the same experiments as those to be collated (i.e., the experiment descriptors file should not contain any extra rows representing experiments which are not represented in the expression data, nor should any experiment which are present in the expression data be missing from the experiment descriptors file). For multi-chip sets, each row should represent an entire set of arrays performed with the same sample, not a separate row for each individual array.

Each succeeding column contains a descriptor variable which may be used for labeling purposes, for identifying reverse fluor arrays, for classification analyses, for identifying replicate experiments, for matching between paired experiments, or for specifying the plotting order of the experiments when clustering genes. The user can create as many columns of the experiment descriptors file as he/she finds useful for classifying the arrayed samples. There should be no empty columns between the experiment descriptor columns.

If the dataset contains reverse fluor experiments, then select this checkbox, and specify the column in the experiment descriptor sheet and the labels in this column which

indicate the reverse fluor arrays. The log-ratio will be computed as log(green/red) instead of log(red/green) for the reverse fluor arrays.

## Special data formats

BRB-ArrayTools can collate data from two special formats for Affymetrix data, from a special format for data archives downloaded from the NCI mAdb database, and also from a data in GenePix format. For data in these formats, BRB-ArrayTools offers "shortcut" **Special format** collation dialogs which allow users to bypass some of the specification fields required in the **Data import wizard**.

Affymetrix data can be exported from MAS 4.0 or 5.0 as probeset-level data (e.g., signals and detection calls from the CHP data) in tab-delimited text files, or imported directly into BRB-ArrayTools from the CEL files. Either of these formats can be imported in BRB-ArrayTools using the **Special format** collation dialogs. Tab-delimited probeset-level data files can be collated as a single-chip type or multi-chip set, whereas CEL files currently can only be collated as a single-chip type. However, probeset-level expression summaries which have already been exported from other software such as BioConductor or dChip should be imported as text files using the standard **Data import wizard**.

## Collating Affymetrix data from CHP files exported into text format

To import data from CHP files which have been exported from MAS 4.0 or 5.0, go to the **ArrayTools → Collate data → Special format: Affymetrix GeneChips → Probeset-level data** menu item.

The CHP files should be exported from MAS 4.0 or 5.0 as tab-delimited text files in either the "horizontally aligned" or "separate files" input data formats as previously described.

### Single chip type

The single chip type refers to the situation when all the chips which were hybridized were of the same type and format, containing the same probesets. Experiments which used only chip 'A' of a multi-chip set can also be considered as a single chip type experiment.

If data is exported in the horizontally aligned input data format, then a Pivot Table should be created containing the absolute analysis data from all experiments that were performed within that chip type, and the pivot table should be exported as a tab-delimited text file.

If the data is exported in the "separate files" format, then the Metrics Table data from each CHP file should be exported as a separate tab-delimited text file in a data folder devoted exclusively to expression data files. The files may contain miscellaneous rows before the column header row, but all files must have the exact same format (and same

number of miscellaneous rows before the column header row). Each file should contain data from one chip or array, and the names of the separate files (without the ".xls" or ".txt" file extensions) will be used as experiment names in the collated project workbook.

**In all of the following formats, the "Probe Set Name", "Signal" and "Detection" are the only required columns from MAS 5.0 data, and the "Probe Set Name", "Avg Diff" and "Abs_Call" are the only required columns from MAS 4.0 data. All other columns are optional.**

BRB-Arraytools can automatically recognize the data format of files that have been exported directly from MAS 4.0 and MAS 5.0. However, users who choose to edit the data columns and column headers of their files should follow the format described below, in order for BRB-ArrayTools to automatically recognize the data format. If the column header row is not in any of the following formats, then BRB-ArrayTools will not be able to parse the files automatically. In that case, the user will need to use either the 'horizontally aligned' or 'separate files' collating dialogs (for a single array type) or the "multi-chip sets" collating dialog (for multi-array sets), in order to collate the data.

<u>Expression data in a "horizontally aligned" file:</u>

1. <u>For expression data output from a Pivot Table in MAS 5.0:</u>

| Probe Set Name | Description | Exp1_ Signal | Exp1_Detection | Exp2_Signal | Exp2_Detection |
|---|---|---|---|---|---|
| 1053_at | Human replication | 234 | P | 456 | P |
| 1773_at | Human farnesyl- | 123 | P | 39 | P |

2. <u>For expression data output from a Pivot Table in MAS 4.0:</u>

| Probe Set Name | Description | Exp1_Avg Diff | Exp1_Abs_Call | Exp2_Avg Diff | Exp2_Abs_Call |
|---|---|---|---|---|---|
| 1053_at | Human replication | 234 | P | 456 | P |
| 1773_at | Human farnesyl- | 123 | P | 39 | P |

For the above two formats, the prefixes Exp1 and Exp2 will be used as experiment names in the collated project workbook. You may use any other experiment names, but do not use the following special characters "\ / : * ? " < > | ." which have special meanings in Windows. The "Probe Set Name" column label may also be given as simply "Probe Set". The detection (Abs_call) column should be right next to the signal (Avg Diff) column for each experiment.

<u>Expression data in a "Separate Files" format:</u>

1. <u>For expression data output from a Metric Table in MAS 5.0:</u>

| Probe Set Name | Description | Signal | Detection |
|---|---|---|---|
| 1053_at | Human replication | 234 | P |
| 1773_at | Human farnesyl- | 123 | P |

2. <u>For expression data output from a Metric Table in MAS 4.0:</u>

| Probe Set Name | Description | Avg Diff | Abs_Call |
|---|---|---|---|
| 1053_at | Human replication | 234 | P |
| 1773_at | Human farnesyl- | 123 | P |

3. In addition to the above mentioned file formats, the following file format from the NCBI portal will also be recognized:

| ID_REF | VALUE | ABS_CALL |
|---|---|---|
| AFFX-MurIL2_at | -896 | A |
| AFFX-MurIL10_at | 682 | A |

4. In addition to the above mentioned file formats, the following file format from mAdb will also be recognized:

| AffyId | Signal | Detection Call |
|---|---|---|
| A28102_at | 7.002252579 | A |
| AB000114_at | 4.121015549 | A |

Gene annotations file format:

The gene annotations file is optional. If you have probe set annotation columns other than the Description column, please place them in a separate tab-delimited gene annotation file and use the following column header labels. For multi-chip sets, please place your entire probe set annotation information for all chip types in just one file.

ProbeSet - the name of the probe set.
Name - the description or title of the probe set.
UnigeneID - the Unigene Id.
Symbol - the gene symbol.
Accession - the GeneBank Id or source of the sequence.
Chromosome - the location of the gene in the chromosomal map.

Experiment descriptors file:

The experiment descriptors file should follow the same format as described in the previous section Experiment descriptors. The first row of the experiment descriptors file should be a column header row, and each subsequent row should represent an individual array. The first column of the experiment descriptors file should contain the experiment labels. When the expression data is horizontally aligned in one file, then the order of the rows in the experiment descriptors file will be assumed to correspond to the order of the array data blocks in the horizontally aligned expression data file. When the expression data is in a separate file for each array, then the experiment labels in the first column of the experiment descriptor sheet must correspond to the expression data filenames (without the ".xls" or ".txt" file extensions).

## Importing Affymetrix data from text or binary CEL files

For Affymetrix data, BRB-ArrayTools now also accepts expression data in the CEL file format. However, the user may only input CEL files of a single chip-type, not as a multi-chip set. The Affymetrix CEL file collation in v3.6, currently has the following options namely the MAS5.0, RMA, almostRMA and the GC-RMA methods to compute probeset summaries. All of the above methods use Bioconductor packages to read the .CEL files and compute the corresponding probeset summaries. BioConductor(http://www.bioconductor.org/) is an open source and open development software project for the analysis and comprehension of genomic data implemented in R.

The MAS5.0 probe set summaries are computed from .CEL files using the justMAS function in the 'simpleaffy' library from Bioconductor. The algorithm is implemented as described in Affymetrix's 'Statistical Algorithms Reference Guide' - see http://www.affymetrix.com, and in Hubbell et al. (2002) Robust Estimators for expression analysis. Bioinformatics 18(12) 1585-1592).

The RMA probe set summaries are computed using the "affy" package .The "RMA" method can be summarized as a three step approach that uses a background correction on the PM data (Perfect Match), then applies a quantile normalization and finally summarizes the probe set information by using Tukey's median polish algorithm. References: RA Irizarry, et.al. "Summaries of Affymetrix GeneChip probe level data" Nucleic Acids Research, 2003, vol.31, No.4.

If there are less than 100 .CEL files, BRB-ArrayTools performs the RMA method to compute probe set summaries. The RMA option uses all the arrays simultaneously to compute the normalization and probe set summaries. However, for large number of arrays(greater than or equal to100), BRB-ArrayTools uses the almostRMA method which is a more memory efficient method. This method uses a random subset of a 100 arrays to develop the quantile normalization and probe effects which are then applied to all the arrays in the dataset. Additionally, these quantiles and probe effects obtained from the subset of arrays are stored under the project folder in a sub-folder called 'almostRMA'( Reference: Darlene Goldstein Bioinformatics 22:2364,2006).

The GC-RMA probe set summaries are computed using the "gcrma" package from Bioconductor.The gc-rma method adjusts for background intensities for Affymetrix data which include optical noise and non-specific binding.The gcrma function converts the background adjusted probe intensities to expression measures using the same normalization and summarization method as RMA.

To import Affymetrix CEL files, place the CEL files in a separate data folder, and make sure all the CEL files within the folder are of the same chip type. Then, go to the **ArrayTools → Import data → Data import wizard** menu item. This utility requires the BioConductor 'affy' package to be installed as an R library. If you do not already

have this package installed, then you will be prompted to install it before proceeding. After you click OK, the importing process will launch.

**Note about collating Affymetrix CEL files**

Please be aware that the collation of CEL files is a very memory-intensive process, and may take several hours to run for a large number of chips. You may wish to check your computer's memory capacity, and allow sufficient time if collating a large number of chips.

## Importing Affymetrix Gene ST1.0 .CEL files

You can import Affymetrix ST-Array by clicking ArrayTools-->Import data--> Affymetrix ST-Array Importer. There are three types of arrays that are supported in BRB-ArrayTools are namely Affymetrix Human Gene 1.0 ST Array, Mouse Gene 1.0 ST Array and Rat Gene 1.0 ST Array. Please refer to Affymetrix website for more details. Please note, that this importer does not support Affymetrix Exon arrays. To import your .CEL files, it should be placed in a separate folder, which contains only the .CEL files. After you browsed for your data folder containing the .CEL files, and selected the organism type, the program will attempt to download and install the necessary custom CDF file and the 'aroma.affymetrix' package from the from aroma-project.org .The custom CDF file will be saved to the "ArrayTool" installation folder (default path is C:/Program Files/ArrayTools/Misc). Similarly, other supporting packages such as 'Biobase' etc get downloaded from Bioconductor. The .CEL file processing is time intensive. It can take up to several minutes per file. The annotations for Affymetrix ST-Array are obtained from Bioconductor. For more details about those annotation packages, please refer to http://www.bioconductor.org/packages/release/data/annotation/.

## Importing RNA-Seq data outputted from Galaxy web tool

Starting from ArrayTools v4.3, RNA-Seq data pre-processed and outputted from the Galaxy web tool (https://main.g2.bx.psu.edu/) in tab-delimited .txt file format can be imported into BRB-ArrayTools for further statistical analysis of gene expression. The user needs to run Galaxy tools to obtain the FPKM estimates for each sample. Upon clicking on ArrayTools-->Import data--> Data Import wizard, a dialog form will pop up ask for the data type. From the dropdown list the user can select the "RNA-Seq Data from Galaxy" as the data type and then browse for the file containing the FPKM estimates to continue. SOURCE annotation can be run at the end of importing.

## Collating data from an NCI mAdb archive

BRB-ArrayTools has a collating interface that allows the National Cancer Institute Advanced Technology Center users to easily collate their data that has been downloaded from the "mAdb" website as a zipped archive. To import "mAdb" data, go to the **ArrayTools → Import data → Data import wizard** menu item. Currently, this collating interface is only implemented for dual-channel cDNA data, and Affymetrix data from a single chip type. The "mAdb" collating interface will be modified in the future to handle Affymetrix multi-chip data as well. For now, however, if the user has an "mAdb" archive that contains Affymetrix multi-chip data, then the user will need to use either the **Multi-chip sets** or **Affymetrix GeneChips** collating dialog.

With the "mAdb" collating interface, the user only needs to browse for the folder that contains the unpacked "mAdb" archive, and for dual-channel data, specify whether or not the genes are aligned and whether or not the data contains reverse fluor experiments. If the data contains reverse fluor experiments, then the user should first edit the `array_descriptions` file by entering a reverse fluor indicator in column "G", in which reversed arrays are denoted by the label "Yes".

The zipped archive should be unpacked before collating. When unpacking the archive, please check to make sure the directory structure of the archive has been preserved.

When the data for each array is in a separate file, there should be two files

```
array_descriptions_xxx_xxxxxx.xls
gene_identifiers_xxx_xxxxxx.xls
```

and a folder

```
array_data_xxx_xxxxxx
```

which are unpacked into the same directory. The `array_data_xxx_xxxxxx` folder contains all the expression data files, where each file contains the expression data for a single array.

Other software developers may also create data archives that are similar in format to the "mAdb" data archive, which can then be easily imported by their users into BRB-ArrayTools using the **NCI Microarray Database (mAdb) archive** collation dialog. For documentation on the expected format for the "mAdb" data archive, please refer to the separate document in the `Doc` folder of the `ArrayTools` installation folder entitled mAdb archive format.doc.

## Collating GenePix data

BRB-ArrayTools can collate GenePix data with the simplified procedure which will recognize this format and enter all necessary columns automatically as compared to the same collation when it is performed in the data import wizard. To import “GenePix” data, go to the **ArrayTools → Import data → Data Import Wizard** from the menu item. User then will enter the data files folder location and experiment descriptor file (or use an option to create descriptors automatically from the files names) and will run the collation. Additional options can be set to flip the ratios on reverse fluor arrays, add extra gene identifiers (gene names and ids are entered automatically from the main data files), and average duplicates within arrays. For dual channel data,

Red intensity - "F635 Mean" or "Ch1 Intensity (Mean)"
Red background - "B635 Median" or "Ch1 Background (Median)".

Green intensity - "F532 Mean" or "Ch2 Intensity (Mean)".
Green background - "B532 Median" or "Ch2 Background (Median)"

For single channel data,

Signal - "F635 Mean".

We also import other important fields:

Unique ID - the "ID" column.
Spot Flag - the "Flags" column.
PrintTip - the "Block" column.
Spot Size - the "Dia." or "Number of Spot Pixels" column

Background-adjusted intensities are computed from the foreground mean minus the background median (usually this is F635 Mean – B635 Median for the red channel, and F532 Mean – B532 Median for the green channel), and the log-ratio is then computed from the background-adjusted red and green intensities.

## Collating Agilent data

In BRB-ArrayToolsv3.6, dual channel Agilent data can be imported using the Data import wizard. Once, the program identifies that it is from the Agilent system then, the following parameters get automatically imported. The **ProbeName"** as the Unique Id, the**"gBGSubSignal"** as the green background subtracted signal,**"rBGSubSignal"**as the red background subtracted signal, the **"gNumPix” or “rNumPix"** as the total number of pixels for Spot Size and the **"gIsFeatNonUnifOL” or “rIsFeatNonUnifOL"** to indicate if a feature is a population outlier or not and is used as the Spot Flag parameter. For annotations either alongside expression data or in separate file, the “ProbeName" gets imported as the Unique ID",”gb" as the GenBank Id, the "ug" as the Unique Gene Cluster ID and the "GeneName" as the Gene Symbol.

For single channel Agilent data, **"gBGSubSignal"** will be imported as the signal column

## Collating Illumina expression data

The Illumina expression data can be imported using the Data import wizard. In order to import data using Data Import Wizard, the users need to export their Illumina data through BeadStudio software. As BeadStudio can output both probe profile and gene profile, the probe profile is preferred. For each array, columns with header containing "AVG_Signal" and "BEAD_STD*" (where * can be any characters) are required to be exported. For users who would like to apply variance stabilizing transformation (VST) through the "lumi" package, two additional columns with headers "beadNum" and "Detection" are also required in the BeadStudio/GenomeStudio exported file. If either of these two columns are missing, log2 transformation will be applied instead of VST. In addition, the output data file is required to contain columns with the unique ID column header named ProbeID, PROBE_ID or TargetID. Upon clicking on ArrayTools -> Data Import Wizard, the user needs to select the data type as "Illumina Single channel data" and browse for the data file. Once the program identifies that it is an output file generated using the BeadStudio software, data get imported by the "lumi" R package available at Bioconductor. The software asks users whether they wishe to lumi package to normalize the data. If non-normalized raw data are imported, they undergo variance-stabilizing transformation, followed by normalization. The users have 5 options to normalize the data: 1) Robust spline normalization (RSN); 2) Loess normalization; 3) Quantile normalization; 4) Rank Invariant normalization and 5) SSN (simple scaling normalization). We recommend users to export non-normalized raw data so as to use the lumi R package to transform and normalize the data. However, if the output data exported from the BeadStudio software are already normalized, the users should answer "No" to the question about using lumi package to normalize the data. .
Starting from v4.2, the users have more options of annotating the Illumina data imported through Data Import Wizard. In order to meet the requirement of different annotation options, users need to make their decisions at the importing step. Three options are available for Illumina data annotation: 1) Custom annotation with the user's own gene identifiers file; 2) SOURCE annotation.
and 3) Annotation with the annotation package associated with "lumi" package.
The software asks whether the user wants to annotate the probes through "lumi". If the user answers "Yes" to this question, the software informs the user that the ProbeID or TargetID will be converted to NuID that is a unique ID specifically used for annotation through the lumi package. The annotation packages can be downloaded through Bioconductor and used for annotation by the "lumi" package (option #3). SOURCE annotation cannot be used if the original Unique ID has been converted to NuID.
If the user answers "No" to the above question about lumi annotation, the software directs the user to browse for a gene identifiers file. The user cannot select the radio button "The Identifiers are stored alongside the expression data", however, the same data file exported from BeadStudio can be selected as the Gene Identifiers file if some annotation columns such as Gene symbol or Entrez Gene Ids have been exported along with the expression data. This file can directly be used for creating custom annotation (annotation option #1), if the user checks the box "Use these gene ids for annotation,

instead of using data from SOURCE database". The "Gene Symbol" column is required for this option, it is a requirement for gene annotation in BRB-ArrayTools. Alternatively, if the user does not check the box, custom annotation will not be conducted. The user can pick one of the columns in the gene identifiers file as query key for SORUCE annotation at the end of importing (option #2). Several gene identifiers can be used as the query key for SOURCE annotation, including Symbol, Accession number, Entrez Gene Id, and Probe Ids starting with "ILMN_".

## Collating Illumina methylation data

Staring from ArrayTools v4.3, the Illumina methylation data can be imported using the Data import wizard. Currently the following three chip types are supported: 1) IlluminaHumanMethylation27k; 2) IlluminaHumanMethylation450k and 3) GGHumanMethCancerPanelv1. The raw data file is a tab-delimited .txt file outputted from either BeadStudio/GenomeStudio software. It is required to have the following columns: 1) TargetID column and 2) The AVG_Beta column for each array. The beta value represents the proportion of methylated signal intensities among all intensities (methylated and unmethylated intensities) in each probe. It is always a number between 0 and 1. In addition, the file could contain columns for the signal intensity data of unmethylated and methylated probes, such as Signal_A and Signal_B, Signal_CY3 and Signal_CY5, or Signal_Red and Signal_Grn, for all samples. Upon clicking on ArrayTools -> Data Import Wizard, the user needs to select the data type as "Illumina methylation data" and browse for the data file. Once the program identifies the "TargetID" and "AVG_Beta" columns, a form pops up to let the user select the chip type. If the user has either the Infinium Human methylation 27k or 450k chip, and the methylated/unmethylated signal data are available in the data file, data get imported by the "lumi" R package available at Bioconductor. The software continues to ask the user whether he/she wishes to use the "lumi" package to normalize the data. If the user answers "Yes" to the question, data will be quantile color balance-adjusted, quantile normalized and then converted to the M values (log2 ratios of methylated over unmethylated normalized signal intensities) using functions in the "lumi" package, otherwise data will be converted to log2(beta/(1-beta)) values. If the raw data file does not contain the signal intensities of methylated and unmethylated probes, or if the chip type is Golden Gate based, data will be read in using the "methylumi" package, and will be converted to log2(beta/(1-beta)) values. No matter what chip type it is, or whether normalization has been applied, the processed data will be treated equivalent to log2 ratio data.
For Illumina methylation data, there are 3 different options of annotation: 1) Annotate probes with the user's own gene identifiers file; 2) SOURCE annotation.
and 3) Annotate probes with the annotation package available at Bioconductor. For options 1) and 2) the user is required to browse for their gene identifiers file, or specify the Gene identifier columns present alongside with the raw data file.

### Collating from NCBI GEO Import Tool

This function allows you to automatically import a GDS dataset from the NCBI Gene Expression Omnibus (GEO) database into BRB-ArrayTools. To import a GEO dataset into BRBArrayTools you only need to provide the GDS number which can be retrieved from the following Entrez interface: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gds&term=all[filter]. Additionally, you can browse the above webpage to see the available datasets. The Import Tool can process ONLY GEO database entries with a GDS number.

The Importer first creates a GDS folder associated with the GDS number that the user entered, it then downloads the GDS data file into the data folder. From the GDS file, it finds the GPL number and downloads and saves the corresponding GPL annotation file into the folder. From GDS file, it extracts the Subset Record such as Sample_id, Description, Protocol to create the Experiment Descriptor file. To import the data into BRB-Arraytools, a project folder called 'GDSXXXX-Project is automatically created. This Project Folder contains all the files BRB-Arraytools needed to analyze the array data. For Affymetrix Data, the annotations are imported from BRB's Linus server, using a chip type specified by the user.

References:

1: Barrett T, Troup DB, Wilhite SE, Ledoux P, Rudnev D, Evangelista C, Kim IF, Soboleva A, Tomashevsky M, Edgar R.
NCBI GEO: mining tens of millions of expression profiles--database and tools update
Nucleic Acids Res. 2006 Nov 11; [Epub ahead of print]

2: Edgar R, Domrachev M, Lash AE.
Gene Expression Omnibus: NCBI gene expression and hybridization array data repository
Nucleic Acids Res. 2002 Jan 1;30(1):207-10

## Output of the collating step

### Organization of the project folder

During the collating step, BRB-ArrayTools will create a **project folder** either within or alongside the folder that contained the user's original raw data files. This project folder will contain the **collated project workbook**, as well as supporting files needed by the collated project workbook. The new project folder will contain a `BinaryData` folder, and possibly also an `Annotations` or `Output` folder. The `Annotations` folder is created if the user chooses to lookup gene annotations from the Stanford SOURCE database. Some BRB-ArrayTools analyses may also produce various output files that will automatically be written to the `Output` folder. For example, all the plots that

appear onscreen in the **Cluster viewer** page during cluster analysis are also automatically saved into the `Output` folder, and may be subsequently edited by the user for publication.

Please note that *only one data project should be collated into a single project folder*. Please use a separate project folder for each collated project, and only re-use an existing project folder if you intend to overwrite the existing collated project within that folder. If you are collating into an existing project folder, and the project folder already contains an existing `Annotations` folder, then the gene identifiers in the existing `Annotations` folder will be compared against the gene identifiers in your new collated project. If the set of gene identifiers in the existing `Annotations` folder match up exactly with the set of gene identifiers in your new collated project, then you will be given the option of importing the existing gene annotations into your new collated project workbook, so that you will not need to lookup the Stanford SOURCE database to get those annotations.

## The collated project workbook

The collating procedure produces a collated project workbook. This is an Excel workbook containing three worksheets labeled **Experiment descriptors**, **Gene identifiers**, and either **filtered log-ratio** (for dual-channel data) or **filtered log intensity** (for single-channel data). All other data variables, such as the red and green background-adjusted signals and raw log-ratios (for dual-channel data), the raw log-intensities (for single-channel data), the spot flag (or Detection Call, for Affymetrix data), and the spot size, are considered auxiliary variables and will be written to separate text or binary files within the project folder.

The rows in the **gene identifiers** worksheet correspond to the rows in the filtered log-ratio (or filtered log intensity) worksheet. The last column of the gene identifiers and filtered log-ratio (or filtered log intensity) worksheets contains an internal filtering variable called "Filter" containing the labels TRUE or FALSE. To change the filtering criteria for any given analysis, the user should *not* manually edit the "Filter" column in these worksheets, but should set the filtering criteria by clicking on the **Re-filter the data** menu item. Re-filtering the data will cause BRB-ArrayTools to automatically adjust the TRUE/FALSE labels in the "Filter" column according to the filtering criteria chosen.

The **experiment descriptors** worksheet of the collated project workbook contains an *editable copy* of the experiment descriptors file which was collated with the data. This copy can be edited by the user to add descriptors to be used in later analyses, as in the example below:

BreastSamples.xls

| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | ExpId | Type | Site | Array-order | | | | | |
| 2 | BC16 | Tissue | Breast | 2 | | | | | |
| 3 | BC2 | Tissue | Breast | 3 | | | | | |
| 4 | BC2_Lymph_Node | Tissue | Lymph Node | 4 | | | | | |
| 5 | BT-549_CL5013_BREAST | Cell-line | Breast | 5 | | | | | |
| 6 | HS_578T_CL5006_BREAST | Cell-line | Breast | 6 | | | | | |
| 7 | MCF7_CL5001_BREAST | Cell-line | Breast | 7 | | | | | |
| 8 | MCF7A-repro | Cell-line | Breast | 8 | | | | | |
| 9 | MCF7D-repro | Cell-line | Breast | 9 | | | | | |
| 10 | MDA-MB-231_CL5005_BREAST | Cell-line | Breast | 10 | | | | | |
| 11 | MDA-MB-435_CL5011_BREAST | Cell-line | Breast | 11 | | | | | |
| 12 | MDA-N_CL5012_BREAST | Cell-line | Breast | 12 | | | | | |
| 13 | Normal_Breast | Tissue | Breast | 1 | | | | | |
| 14 | T-47D_CL5014_BREAST | Cell-line | Breast | 13 | | | | | |
| 15 | | | | | | | | | |

Experiment descriptors / Gene identifiers / Filtered log ratio

The **Array-order** descriptor variable, for example, might be used to specify the order of experiments to use when plotting lineplots of gene cluster expression.

The **gene identifiers** worksheet of the collated project workbook contains the gene identifiers whose columns had been specified in the **column format** section under the **gene identifiers** page of the collating dialog box.

BreastSamples.xls

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | Unique i | Clone | Description | GB acc | Defined functions | Filter |
| 4 | 51 | 60218 | EST | T39165, T40450 | | TRUE |
| 8 | 55 | 60955 | EST | T39626, T40701 | | TRUE |
| 15 | 62 | 53049 | EST | R16185, R15749 | | TRUE |
| 32 | 79 | 71644 | EST | T57978, T57896 | | TRUE |
| 54 | 101 | 76539 | MGP Matrix Gla protein Chr.12 | T60828, T60778 | | TRUE |
| 57 | 104 | 78275 | Nuclear antigen Sp100 | T50942, T50780 | | TRUE |
| 59 | 106 | 78736 | EST | T60389, T61888 | | TRUE |
| 63 | 110 | 79378 | Homo sapiens 195 kDa cornified envelope prec | T57706, T57667 | | TRUE |
| 66 | 113 | 80277 | EST | T65704, T64351 | | TRUE |
| 69 | 116 | 81604 | AOAH Acyloxyacyl hydrolase (neutrophil) Chr.7 | T66021, T65864 | | TRUE |
| 74 | 121 | 84203 | Cytochrome P450, subfamily IIA (phenobarbital- | T72786, T72911 | Drug_metabolism | TRUE |
| 77 | 124 | 84545 | EST | T73940 | | TRUE |
| 80 | 127 | 86044 | PFKFB1 6-phosphofructo-2-kinase/fructose-2,6- | T62734, T62884 | | TRUE |
| 81 | 128 | 85924 | Cytochrome P450, subfamily IIA (phenobarbital- | T73132, T73006 | Drug_metabolism | TRUE |

Experiment descriptors / Gene identifiers / Filtered log ratio

If the user has chosen to annotate his or data using the defined genelists, then the gene identifiers worksheet will also contain a column labeled **Defined genelists**, indicating if the genes matched any of those listed in the **user defined gene lists**.

The click-arrows in each cell of the header row may be used to search for specific values of each variable. Column-widths may be adjusted by hovering the cursor between the columns in the lettered column header bar and dragging.

The **filtered log ratio** (or **filtered log intensity**) worksheet contains the log-ratios (or log intensities) *after* the specified filtering levels have been applied. These are the data values that will be used for all analyses except the scatterplot of experiment v. experiment, where the log of the channel intensities can also be plotted for dual-channel data. The first column of the worksheet contains the primary gene id, which matches the

gene id listed in the first column of the gene identifiers worksheet. Each succeeding column contains the log-ratios obtained from a single experiment. The highlighted yellow columns at the end of the dataset are used internally by BRB-ArrayTools for gene-filtering purposes. The user should not edit this worksheet in any way. BRB-ArrayTools will automatically re-filter this worksheet, if necessary, before running each analysis.

BreastSamples.xls

| | A | B | C | D | E | M | N | O | P | Q | R |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Unique i | BC16 | BC2 | BC2_Ly | BT-549_ | Normal_ | T-47D_C | P-Value | Rank | Missing | Filter |
| 4 | 51 | | | | 0.4288 | -2.80023 | -0.88536 | 0.003498 | 2066 | 8 | TRUE |
| 8 | 55 | 1.97728 | 1.229482 | | 1.1227 | -1 | 2.472171 | 0.007142 | 2255 | 1 | TRUE |
| 15 | 62 | -1.51706 | -0.53343 | -1.50815 | -3.4374 | -1.5736 | -0.78165 | 0.005309 | 2168 | 5 | TRUE |
| 32 | 79 | | -3.06413 | -0.14975 | 0.4116 | -0.03116 | 0.073462 | 5.62E-11 | 468 | 6 | TRUE |
| 54 | 101 | 2.383922 | 3.406278 | 3.192199 | 1.1458 | 1.823643 | 2.51405 | 0.001004 | 1776 | 0 | TRUE |
| 57 | 104 | 1.507907 | -0.18521 | 1.970129 | 0.0437 | -0.1947 | -0.91867 | 2.46E-07 | 812 | 1 | TRUE |
| 59 | 106 | -0.50657 | 0.972293 | -1.01707 | -3.077 | 0.258663 | -0.68296 | 2.5E-08 | 701 | 2 | TRUE |
| 63 | 110 | 0.956352 | 2.626515 | 1.871786 | -2.7784 | 0.952251 | 0.27711 | 1.99E-21 | 132 | 2 | TRUE |
| 66 | 113 | | 1.956057 | -0.29418 | 1.4094 | -0.7422 | -1.28011 | 0.007637 | 2270 | 1 | TRUE |
| 69 | 116 | | | | | | 0.25573 | 0.000569 | 1661 | 9 | TRUE |
| 74 | 121 | | 6.030974 | 3.040414 | -1.219 | 1.143591 | -0.24345 | 4.31E-12 | 395 | 6 | TRUE |
| 77 | 124 | | 2.399407 | | 1.022 | -2.60407 | -1.07836 | 5.56E-05 | 1307 | 8 | TRUE |
| 80 | 127 | 1.853253 | 1.121991 | -2.07155 | 0.2721 | -1.72016 | 0.995369 | 0.001645 | 1875 | 5 | TRUE |
| 81 | 128 | 0.58823 | 3.974842 | 1.198367 | -0.2358 | -1.21042 | -0.01749 | 7.88E-11 | 482 | 3 | TRUE |

Filtered log ratio

# Filtering the data

The filtering criteria are specified during the data collating step, and may be changed prior to each analysis. The dialog box for each analysis has a **Filter** button on the bottom, which can be used to specify or change the filtering criteria for all subsequent analyses.

Although the filtering criteria may be changed for each analysis, it may be important to check the filtering criteria before collating the data *if* the data contains replicate spots for some genes in the array. This is because the **spot filtering** is applied *before* averaging the replicate spots, and replicate spots are *not* re-averaged after the collating step has been completed, even if the user changes the filtering criteria.

The **Filtered log ratio** (or **Filtered log intensity**) worksheet of the collated project workbook will reflect the filtering actions which have been selected. With the exception of replicated spots, which are filtered before averaging, the supporting files that are located in the project folder will *not* reflect the filtering actions that have been selected.

The order of operations for filtering the data is that spot filters are applied first, then data normalization, then truncation of extreme values, then gene screening.

## Spot filters

Spot filtering refers to filters on spots in individual arrays. Spot filtering is used for quality-control purposes, to filter out "bad" spots. Unlike gene screening, spot filtering does not filter out the entire gene (row), but replaces the existing value of a spot within any given array with a missing value.

### Intensity filter

For dual-channel data, a spot may either be filtered out (excluded) from the analysis, or thresholded (with one signal set equal to the specified minimum) in the analysis. Using the appropriate filtering option, a spot on any array that has a red signal less than the specified minimum and/or a green signal less than the specified minimum will have its log-ratio value filtered out on the **filtered log ratio** worksheet. Using the thresholding option, a spot on any array which has BOTH signals less than the specified minimum will be filtered out, but a spot that has only ONE signal less than the specified minimum will have the below-minimum value set equal to the specified minimum, in computing the log-ratio value that is shown in the **filtered log ratio** worksheet. The intensity filter is applied to the *background-adjusted* red and green signals.

For single-channel data, a spot on any array, which has a signal intensity less than the specified minimum, may either be filtered out or thresholded on the **filtered log intensity** worksheet. Please note that the signal intensity values have already been automatically

thresholded to 1 by default, unless the "Do not threshold to 1 (e.g., CodeLink)" checkbox was selected at the time of collating. If the "Do not threshold to 1" option was selected at the time of collating, then negative and 0 signal intensities have already been set to missing, and signal intensities in the interval between 0 and 1 have been left as-is. Please note that the "Do not threshold to 1" option is IRREVERSIBLE once the data has been collated! Once the negative or 0 values have been set to missing, they can never be thresholded again to 1 by re-filtering the data. Likewise, once the values less than 1 have been thresholded to 1, the negative or 0 values can never be separated from the values in the interval between 0 and 1 and be subsequently set to missing. In order to change the "Do not threshold to 1" option, the data must be re-collated.

### Spot flag filter

The spot flag filter can contain both numeric and character values. The user may specify a numeric range outside of which a numeric flag is considered to be "excluded", and/or specify a list of flag values denoting the "excluded" values. When the flag filter is on, a spot on any array which has an "excluded" flag value will be filtered out on the **filtered log ratio** (or **filtered log intensity**) worksheet. The flag field is optional.

For Affymetrix users, a Detection Call column can be designated as the spot flag column at the time of collating, allowing users to filter out expression values that have an "A" (Absent) call. Additionally, the spot flag column (or Detection Call column, for Affymetrix users) is also used by the Percent Absent Filter, to filter out spots (or probesets) with a large percentage of expression values that have a spot flag (or Detection Call) value of "A".

### Spot size filter

The spot size field is optional, but when the spot size filter is on and the spot field is present, then a spot on any array which has a spot size less than the minimum value will be filtered out on the **filtered log ratio** (or **filtered log intensity**) worksheet. For Affymetrix data, the number of pairs used to compute the signal can be used as a surrogate for the spot size measurement.

### Detection call filter

For data collated using the **Affymetrix GeneChips** collating interface, the Detection call filter is a special case of the spot flag filter mentioned above, allowing users to filter out expression values based on the Detection call. The Detection data column is also used by the Percent Absent Filter to filter out probesets with a large percentage of Detection Calls that have a value of "A".

## Transformations

A logarithmic (base 2) transformation is applied to the signal intensities (for single-channel data) or intensity-ratios (for dual-channel data) before they are normalized and truncated. The effect of the data normalization and truncation will be reflected in the **Filtered log ratio** (or **Filtered log intensity**) worksheet of the collated project workbook.

## Normalization

For dual channel data there are currently four normalization options: median normalization, housekeeping gene normalization, lowess normalization,and print-tip group normalization.

### Median normalization

The median-normalization of dual-channel data is performed by subtracting out the *median log-ratio* for each array, so that each normalized array has a median log-ratio of 0.

### Housekeeping gene normalization

The user may specify a housekeeping genelist that will be used for normalization. The housekeeping genes normalization of dual-channel data is performed by subtracting out the *median log-ratio over housekeeping genes* from all the log-ratios on the array.

### Lowess normalization

For dual-channel data, a lowess (or intensity-dependent) normalization option is also available. The median normalization is equivalent to multiplying all of the intensities in one channel of an array by a normalization factor. In some cases it can be advantageous to have a different normalization factor for different intensity levels; i.e., the dye bias may be different for low intensity spots relative to high intensity spots. In the lowess normalization, a non-linear lowess smoother function is fit to the graph of un-normalized log-ratio on the y-axis versus average log intensity (i.e., [log(Red)+log(Green)]/2 ) on the x-axis. This is the so-called M-A plot. The lowess smoother is based on a concatenation of linear regressions for points in overlapping intervals on the x-axis of the M-A plot. This lowess smoother is subtracted from the un-normalized log-ratios for the array in order to obtain the normalized log-ratios. The lowess normalization is much more computationally intensive than the median centering normalization and may require up to 10 seconds per array on some computers using data with 35,000 genes. You may wish to first normalize your data using the default method and then examine M-A scatterplots to determine whether intensity dependent normalization is needed. If most of the points of the M-A plot are distributed equally above and below the 0 value of the y-axis, without a trend over x-values, then intensity based normalization is not needed.

The default span parameter for the lowess smoother is set to 2/5.. This means that 2/5 of the total set of data points will influence the smooth at each value. Larger span values give more smoothness. In previous versions, the default span value had been set to 2/3,

but this was found to create overly-smoothed curves which did not sufficiently capture the trends which were often found in the tails of the M-A plot. The user is able to change the default span value by editing the "LowessSpan" parameter value in the "Preferences.txt" file in the "Prefs" folder of the ArrayTools installation folder. The "Preferences.txt" file is a tab-delimited file in which parameter names are stored in the first column and parameter values are stored in the second column. To modify this option you can go to ArrayTools pull down menu select "Utilities"->"Preference" and change the options. Then choose the button to "Save changes" and "Exit".

## Print-tip Group/Sub Grid normalization

Global normalization calculates a normalization value for the entire array. This method is vulnerable to spatial effects and non-uniformity of the print-tips. Print-tip or sub-grid normalization attempts to adjust for such systematic variation in dual channel experiments. It is less vulnerable to spatial effects and non-uniformity of signal across an array. Print-tip group may also be called grid or sub-grid. For mAdb data it is called Block. The print-tip group data should be placed along side the expression data with one column for each array.

Median Print-tip normalization independently calculates a median value of the log-ratios from the same print-tip group. This value is then applied only to spots from the same print-tip. For example, if an array was produced by a system with 16 print-tips, then 16 normalization values will be used for each array.

Lowess Print-tip normalization uses only the intensity data that are from the same print-tip group to calculate a normalization value. This is value is then applied only to spots from the same print-tip.

## Single channel data normalization

Starting from v4.2, more normalization options have been added for single-channel data. There are four methods available as follows: 1) Quantile normalization; 2) Normalization by user-specified target intensity and percentile; 3) Normalization by reference array and 4) Normalization by array groups.

### Quantile normalization

Quantile normalization is a method that makes the distribution of each array identical to a target distribution (Boldstad et al.). The detailed algorithm is as follows,

1) Order the log transformed intensities of each array.
2) Compute the target distribution by calculating the arithmetic means of the ordered log intensities across all arrays.
3) Assign the values in target distribution back to each array based on the rank of each probe (set), i.e. the highest-ranked probe (set) will take the highest value in

target distribution, and the second highest-ranked probe (set) will take the second highest value in target distribution, as so on.

### Normalization by specified target intensity and percentile

This normalization method is conducted based on each individual array by subtracting a particular number from all the log2 transformed intensities, such that a user-specified percentile of all probe (set) intensities is equal to the target intensity. The detailed algorithm is as follows,

1) For each individual array, let N be the number of probes/probe sets, and let *j* be an index of probes/probe sets running from 1 to N. Let X and Y be user-specified target intensity and percentile, respectively. Let the intensity of a particular probe (set) *j* be $I_j$. Let the intensity at the Yth percentile be $I_Y$.
2) Compute the difference (D) between the log2 transformed intensity at Yth percentile and the log2 (target intensity) by:
   $D = \log2(I_Y) - \log2(X)$
3) The normalized value (M) for each probe (set) *j* is calculated as:
   $M_j = \log2(I_j) - D$

In the normalization dialog form, there is a "Use housekeeping genes only" checkbox, through which the user can specify a housekeeping genelist that will be used for normalization. When this option is checked, instead of the full set of probes (probe sets), only the probes (probe sets) on the housekeeping genelist will be used to determine "D" in the above procedure.

### Normalization by reference array

This normalization method is a combination of "Median normalization of single channel data" and "Housekeeping gene normalization" in versions prior to v4.2, where a reference array needs to be selected so as to be normalized against all other arrays. The normalization is performed by computing a gene-by-gene *difference* between each array and the reference array, and subtracting the *median difference* from the log-intensities on that array, so that the gene-by-gene difference between the normalized array and the reference array is 0.

The user has the option to explicitly select one of the arrays to be the reference array, or ask BRB-ArrayTools to automatically select the "median" array as the reference array. The algorithm which BRB-ArrayTools uses to select the "median" array is as follows:

1) Let N be the number of experiments, and let *i* be an index of experiments running from 1 to N.
2) For each array *i*, the median log-intensity of the array (denoted $M_i$) will be computed.

3) A median M will be selected from the $\{M_1, \ldots, M_N\}$ values. If N is even, then the median M will be the lower of the two middle values.
4) The array whose median log-intensity $M_i$ equals the overall median M will be chosen as the median array.

For this normalization option, the user may also be able to specify a housekeeping genelist, over which the *median difference* will be calculated instead of the full set of probes/probe sets.

### Normalization by array groups

This option allows the users to normalize their data within array groups, or subsets. The user is required to enter the class column that specifies the array groups. For each array group, a median reference array is created based on all the arrays in this group, and each array is normalized against this median reference array using the same algorithm as described in "Normalization by reference array". The user cannot pick the reference array for each group. The housekeep genelist option cannot be applied as well.

### Truncation

This option allows the user to specify a maximum intensity (for single-channel data) or intensity ratio (for dual-channel data) to be used for analysis. Any values greater than the specified threshold will be truncated to the threshold. For truncation of intensity ratios, both the intensity ratios and inverse intensity ratios will be truncated (e.g., an intensity ratio threshold of 64 means that all intensity ratios will be truncated to lie between 1/64 and 64). Truncation is primarily used for dual-channel data, where small denominators can cause intensity ratios to become enormous.

## Gene filters

Gene filtering, unlike spot filtering, is not applied on an array-by-array basis for each gene. Instead, it uses a criterion based on *all* arrays for a given gene, to determine if that gene should be screened out or not. Its purpose is not to filter out "bad" spots, but rather to screen out genes that are not likely to be informative. The last column (labeled **Filter**) on the **Gene identifiers** and **Filtered log ratio** (or **Filtered log intensity**) worksheets are internally used by BRB-ArrayTools to select the genes that pass the screening.

Here the criterion for filtering out a gene is based upon the variability of its log expression values across all arrays after normalization. Filtering low variance genes is not really necessary except for clustering genes, where the computer memory requirements increase rapidly with the number of genes clustered. Several filtering options are available.

## Minimum fold-change filter

Genes that have low variability may be filtered out using the minimum fold-change filter. Here the criterion for filtering out a gene is based upon the percentage of expression values for that gene which have at least a minimum fold-change from the median expression value for that gene. (If the dataset contains 250 or more experiments, then the mean will be used instead of the median for computational efficiency.) The user may specify the minimum fold-change that is required. If less than a specified percentage of expression values meet the minimum fold-change requirement, then the gene is filtered out.

## Log expression variation filter

Alternatively, the filtering can be based on the variance for the gene across the arrays. One can exclude the x% of the genes with the smallest variances, where the percentile x is specified by the user. Or a statistical significance criterion based on the variance can be used. If the significance criterion is chosen, then the variance of the log-ratios for each gene is compared to the *median* of all the variances. Those genes not significantly more variable than the median gene are filtered out. The significance level threshold may be specified by the user. Specifically, the quantity $(n-1)\ Var_i / Var_{med}$ is computed for each gene i. $Var_i$ is the variance of the log intensity for gene i across the entire set of n arrays and $Var_{med}$ is the median of these gene-specific variances. This quantity is compared to a percentile of the chi-square distribution with n-1 degrees of freedom. This is an approximate test of the hypothesis that gene i has the same variance as the median variance.

## Percent missing filter

Here the criterion for filtering out a gene is based upon the percentage of expression values that are not missing and not filtered out by any of the previous spot filters.

## Percent absent filter

Here the criterion for filtering out a gene is based upon the percentage of Absent calls in the Spot Flag or Detection variable. This gene filter may be applied *independently* of the Spot Flag Filter or Detection Call Filter described in the "Spot Filters" section above, though it uses the same Spot Flag or Detection Call variable. For instance, a user may choose to turn *off* the Detection Call Filter in order to preserve all "Absent" expression values as they are, but turn *on* the Percent Absent Filter in order to exclude probesets that are considered unreliable or uninteresting because too many of the expression values were "Absent". In this case, probesets that did not get excluded by the Percent Absent Filter still maintain their expression values even for those values that were "Absent".

### Minimum Intensity filter

New to v4.1-Beta_2, is the criterion for filtering out a gene is only applied for single channel data. It is based upon the intensity value at a specified percentile across all arrays. For any given gene, if its normalized log intensity is lower than the specified minimum log intensity, it will be filtered out.

## Gene subsets

Gene subsetting, unlike gene filtering, is not based on the expression data values for the genes, but rather on the identities of the genes. The purpose of gene subsetting is to select or exclude genes which are known to be interesting or non-informative based on one of the gene labels.

### Selecting a genelist to use or to exclude

The user may select one or more genelists that define a gene subset, and filter the data to include only those genes in the selected subset (or exclude the genes from that subset). See the section below on Defining annotations using genelists for more details on how genelist files are used.

### Specifying gene labels to exclude

The user may choose to exclude genes by specifying a string within one of the gene identifiers as an exclusion criterion. For instance, a user may choose to exclude all empty wells, so the user may choose to exclude all genes with “Empty well” in the Description column of the Gene identifiers worksheet.

### Reducing multiple probes/probe sets to one, per gene symbol

The program provides an option to use all the probes that correspond to a gene or use only the most variable probe measured by the Inter-Quartile Range (IQR) across all the arrays. For single channel data, an additional option to use only the probe set with the largest expression across all the arrays is also available. By selecting this option, the reduced set of genes is used for downstream analysis.

# Annotating the data

## Defining annotations using genelists

Genelists are text files containing lists of genes belonging to the same functional grouping, where the filename indicates the functional grouping that describes the set of genes listed in the file. Genelists used for two different purposes within BRB-ArrayTools: (1) to annotate the genes in the dataset; and (2) to subset the data based on functional groupings.

One purpose of genelists is to annotate the genes (using the **Utilities → Annotate data → Match data against genelists** menu item). If any of the genes in the user's dataset match any of those in the genelists, then the name of the genelist will appear in a column of the **gene identifiers** page in the collated project workbook. For example, if the user has the "Activating transcription factor 3" gene in his dataset with the identifiers CloneID 428348 and Symbol ATF3, then the CloneID would be matched against the user genelists and found to be in the "Example –Proliferation" genelist file, while the Symbol ATF3 would be matched against the CGAP genelists, and found to be in the "CGL:gene_regulation" and "CFL:transcription" genelist files. Thus, an additional column named **defined genelists** would appear in the **gene identifiers** page of the collated project workbook, and the entry under the defined genelists column would be `Example- Proliferation, CGL:gene_regulation, CGL:transcription` for this particular gene. If the user adds or deletes a genelist after the data has already been collated, then the **defined genelists** column in the **gene identifiers** page can be updated by clicking on the **ArrayTools → Utilities → Annotate data using genelists** menu item**.**

Another purpose is to subset the dataset based on functional groupings (using the **Filter and subset the data** menu item, on the **Gene subsets** tab). When the user chooses one or more genelists with which to subset the data, the subset of genes with a match against any of the gene identifiers listed in any of the selected genelists will be chosen for analysis.

Currently there are two basic categories of genelists: user-defined genelists, and genelists that come pre-loaded with BRB-ArrayTools (e.g., CGAP curated genelists, BioCarta pathways, and KEGG pathways).

### User-defined genelists

BRB-ArrayTools allows the user to associate genes with defined functions or pathways through the use of user-defined genelists. User-defined genelists may be stored in two locations:

1. The `Genelists\User` subfolder of the `ArrayTools` installation folder contains genelists which are made visible to *all* projects.

2. Each project folder also contains a `Genelists` subfolder which contains genelists which are made visible *only* to the specific project.

Genelists placed within either of these folders are simply ASCII text files, where the name of the file denotes the function or pathway that is defined by the genes listed in the file.

1. Genelists using a single gene identifier column:

When the genelist contains a single gene identifier type, the first row of the file specifies the type of gene identifier used within the file, and must *exactly* match one of the following pre-defined labels:

```
UniqueID
CloneID
ProbeSet
Accession
UGCluster
Symbol
```

Each succeeding row of the file should contain the gene identifier of a specific gene or clone to be included in the list. For example genelist file named `CellCyclePathway.txt` might contain the following text:

```
Symbol
CDC25A
CDK2
CDK4
CDK6
CDK7
```

The `ArrayTools\Genelists\User` directory currently contains some examples of genelists using this format, which will be used with the `BreastSamples` dataset that is provided with this package. These example genelists are used with the example dataset, and may be deleted at any time by the user.

2. Genelists using multiple gene identifier columns:

Alternately, the genelist file may have a multi-column format, in which each row after the header row represents one gene, and each column represents one type of gene identifier. All the gene identifier types listed in the header line must still exactly match one of the pre-defined labels listed above. For example, the above `CellCyclePathway.txt` might contain the following tab-delimited text instead:

```
UGCluster       Symbol    Accession
Hs.1634         CDC25A         NM_201567
```

```
Hs.19192        CDK2            NM_052827
Hs.95577        CDK4            NM_000075
Hs.119882       CDK6            NM_001259
Hs.184298       CDK7            NM_001799
```

When using such a multi-column genelist format, a particular gene in the dataset is matched against the first unique identifier column (UniqueID or ProbeSet) only, if this column is present in the user defined gene list. Otherwise, it is identified to be of the pathway or function (in this case, the cell cycle pathway) if *any* of the gene identifier columns is found to yield a match.

The `ArrayTools\Pathways\Output\BioCarta` directory contains some examples of some genelists that use this multi-column format.

Some procedures, such as the Class Comparison, Class Prediction, Survival Analysis and Quantitative Traits Analysis tools, will automatically create new user genelists. The name of the genelist produced by these analysis tools is specified by the user before running the analysis. If there is already an existing genelist with the same name, then that genelist will be overwritten. Some plugins also create genelists, though the user usually doesn't specify the name of the genelist in those cases. Beginning with v3.5 of BRB-ArrayTools, the genelists produced by analysis tools and plugins will no longer be used for annotating genes, though they were still be available for defining gene subsets.

### CGAP curated genelists

In the `Genelists` subfolder of the `ArrayTools` installation folder, there is also a `Cgap` subfolder in addition to the `User` subfolder mentioned above. The `Cgap` subfolder contains curated gene lists which have been downloaded from the Cancer Genome Anatomy Project website (http://cgap.nci.nih.gov/Genes/CuratedGeneLists). These HTML files have been included with earlier versions of BRB-ArrayTools, but will be phased out beginning with version 3.5, since these curated genelists are not actively being maintained by the CGAP group.

### Defined pathways

BRB-ArrayTools also provides defined pathway genelists in the `Pathways\Output` subfolder of the `ArrayTools` installation folder. Currently, BRB-ArrayTools provides BioCarta and KEGG pathways which have been obtained from the Cancer Genome Anatomy Project (http://cgap.nci.nih.gov/Pathways). The user may also automatically download pathways and signatures from the Broad Institute/MIT.

## Automatically importing gene annotations

Annotations can be automatically imported by BRB-ArrayTools if the user's dataset contains a standard gene identifier type such as the UniGene cluster, gene symbol, clone, or GenBank accession, or if an Affymetrix chip type has been specified. The imported annotations may contain the descriptive gene name, UniGene cluster, gene symbol, Entrez Gene ID, Gene Ontology, chromosomal location, and other functional annotations.

Projects whose gene identifier column headers contain "UnigeneID", "ClondID", "Symbol", "EntrezID" or "Accession" can be annotated by using the SOURCE annotation tool. This tool is found under the following menu item: **ArrayTools → Utilities → Annotate data → Import SOURCE annotations.** Starting from release v4.1, SOURCE annotation can be run on 8 different organisms (human, mouse, rat, bovine, C. elegans, fruit fly, yeast and Arabidopsis) using Clone Id, GenBank Accession number, Gene Symbol, Entrez ID or UniGene Cluster ID as the query key. In addition, for Agilent data, Illumina data whose Unique ID starts with "ILMN", and Affymetrix data where custom cdf option is not selected, "ProbeID" or "ProbeSet" can also be used as the query key for SOURCE annotation.

Affymetrix data can be annotated by downloading pre-formatted R-packages from Bioconductor (www.bioconductor.org) or from the custom cdf file website(http://brainarray.mbni.med.umich.edu/Brainarray/Database/CustomCDF), if the custom cdf option is selected.Data which is imported as Affymetrix can be annotated by downloading pre-formatted R-packages from Bioconductor (www.bioconductor.org). This automatic annotation appears as an option in the Data import wizard. If the user did not annotate the data at the time of collating or wishes to update the annotations that have already been imported into the project workbook, the user may run the Affymetrix annotations tool at any later time. This tool is found under the following menu item: ArrayTools → Utilities → Annotate data → Import Affymetrix annotations. The Affymetrix annotation R-packages are updated quarterly by Bioconductor core team, so the user may wish to update the local Affymetrix annotations files by running this tool periodically. Affymetrix annotation can only be run on the Affymetrix array chips with available corresponding annotation packages at Bioconductor. A complete list of these packages is stored in the ArrayTools installation folder (default: C:\Program Files\ArrayTools)\Affymetrix\GeneChipNames.txt.

For Illumina expression data, the user can also choose to use the annotation R packages available at the Bioconductor website. In order to use these packages, the Illumina probe ID or Target ID is required to be converted to the "NuID" using the "lumi" package. Once the Ids are converted the SOURCE annotation tool cannot be applied using the converted Ids as the query key. This tool is found under the following menu item: ArrayTools → Utilities → Annotate data → Import lumi annotations. As for the Illumina methylation data, annotation can be conducted using the annotation packages available at Bioconductor in a similar manner to the Affymetrix chips.

The annotations downloaded by either the SOURCE annotations tool or the Affymetrix annotations tool will be stored in a Gene annotations worksheet inside the project workbook and listed as separate text files under project folder's Annotations subfolder.

The gene annotation column listed on the analysis results HTML page contains hyper links to the gene annotation HTML page where available annotation terms are shown for each gene listed in the gene table of analysis results HTML page. The gene annotation terms are from gene annotation text files under the project folder's annotations subfolder, which are created by ArrayTools annotation utility either based on SOURCE online database or Bioconductor annotations R-packages for Affymetrix data. From release v3.7, a new hyperlink "DrugBank" is available among other gene annotation terms on the gene annotation HTML page. DrugBank is a unique bioinformatics/cheminformatics resource that combineds detailed drug data with comprehensive drug target information, available online at http://redpoll.pharmacy.ualberta.ca/drugbank/.

Once user clicks the hyperlink "DrugBank: Query through Gene Symbol", ArrayTools will link to DrugBank's online query page using the specified gene symbol and show the list of drug cards returned in the Web browser.

## Importing gene identifiers for custom annotations

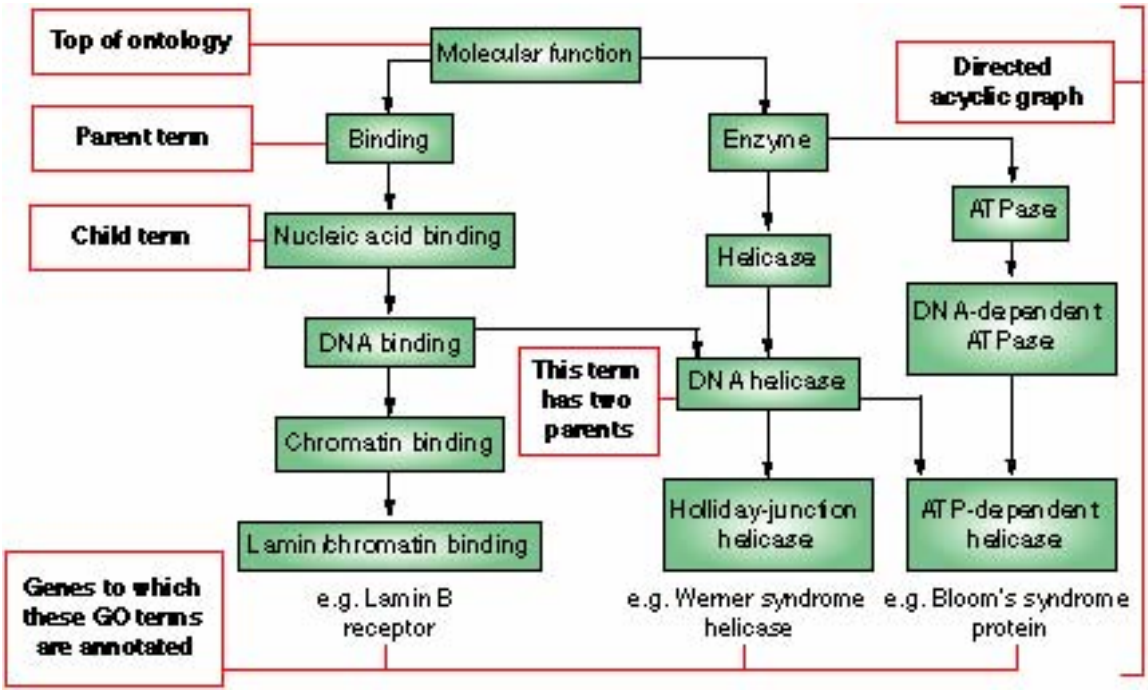
When importing gene identifiers during the data import step, there is a new option added to convert the gene identifiers to gene annotations. By selecting this option, those data sets which have custom annotations can be used as gene annotations for downstream analysis.

## Importing annotations from an existing project with the identical chip type

Starting from v4.3, the annotation information can be imported from an existing project with the identical chip type. The content of the Unique ID/ProbeSet column of the gene identifiers in the current project is required to be identical to the one in the existing project from which the annotation is imported. This can help reduce the redundancy of annotation on multiple projects with the same chip types. However, we do not recommend users to import annotations from a very old project in that they may not obtain the updated annotation information in their newly collated projects. This tool is found under the following menu item: ArrayTools → Utilities → Annotate data → Import annotations from an existing project.

## Gene ontology

Gene Ontology is a structure, controlled vocabulary used to describe gene products in terms of their associated biological processes, cellular components and molecular functions. For more information about Gene Ontology, please browse to http://www.geneontology.org. The vocabularies exhibit a complex structure, represented as directed acyclic graphs (DAGs) or networks. The following is an example from the http://www.geneontology.org/doc/GO.doc.html#process .




If the SOURCE or Affymetrix annotation tools have already been run on the dataset, then BRB-ArrayTools can automatically generate the complete Gene Ontology information for all genes in the dataset using the structure files downloaded from the Gene Ontology database. The Gene Ontology information is used in the Gene Set Comparison tool and the GO observed vs. expected ratio option available for class comparison, survival and quantitative traits analysis tools. From release v3.7, the pre-formatted annotation R-package GO downloaded from Bioconductor (www.bioconductor.org) is used in the Gene Ontology related analysis tools within ArrayTools.

# Analyzing the data

## Scatterplot tools

### Scatterplot of single experiment versus experiment and phenotype averages

It is frequently of interest to plot the log-ratios of one experiment versus the log-ratios of another experiment. Generally, most of the plotted points, which represent genes with nonmissing values in both experiments, will line up along a 45-degree diagonal line. All genes that pass the filtering criteria will be used in the scatterplot, unless a genelist has been selected. In v4.1.0, the scatter plot tools have been modified to be interactive and provide additional options like modifying the arrays plotted on the axes, highlighting genes from a pathway within the plot, selecting up and down regulated genes,linking genes in different plots and exporting genes to a genelist file. To view details of how to run the plots either 2-D and 3-D you can run the video available on http://linus.nci.nih.gov/PowerPointSlides/Scatterplot.wmv

## Scatterplot of phenotype averages

To compare the experiments in one phenotype class versus the experiments in another phenotype class, BRB-ArrayTools provides a tool that plots the average log-ratio within one class on the x-axis versus the average log-ratio within the other class on the y-axis. These averages are taken on a gene-by-gene basis, and each gene is represented by a single point in the resulting scatterplot. The experiment descriptor sheet must contain a descriptor variable containing the class labels to be compared. A blank label in this column means that the corresponding experiment will be omitted from the analysis. For this analysis, the column should contain only two unique class labels, and all labels belonging to other classes should be "blanked" out by the user.

Genes that are differentially expressed between the two phenotypes will fall outside of a pair of outlier lines. These outlier lines can be specified by the user to indicate genes for that the fold-difference between the geometric mean of the expression ratios within each of the two classes is greater than a specified amount.

As with the scatterplot tool for a single experiment versus experiment, similar functionality is available.

# Hierarchical cluster analysis tools

For a given dataset, hierarchical clustering can occur in two directions: either the genes can be clustered by comparing their expression profiles across the set of samples, or the samples can be clustered by comparing their expression profiles across the set of genes. In BRB-ArrayTools, the data can be clustered in either one or both directions independently. The objects to be clustered may be either the genes or the samples.

Hierarchical clustering produces a sequentially nested merging of the genes (or samples) determined by a defined measure of pair-wise similarity or distance between expression profiles. The nested merging is represented by a "dendrogram". At the lowest level of the dendrogram, each gene (or sample) is a member of an individual singleton cluster. At the first step, the genes (or samples) with the two expression profiles most similar to each other are merged into a cluster. Then the next two most similar genes (or samples) are joined as a cluster. At each step, the most similar two clusters (including singleton clusters) are joined to form a larger cluster. This might actually involve the joining of two singleton genes (or samples), merging a gene (or sample) into an existing cluster, or merging two clusters formed at a previous step. The "distance" between two clusters which merge into a single cluster can be read from the scale along side of the dendrogram. Clusters that are merged low on the dendrogram are similar, whereas clusters which are formed by mergers high on the dendrogram may be very heterogeneous. At the top level of the dendrogram, there is a single cluster containing all of the genes (or samples). At the lowest level of the dendrogram the "clusters" are very homogeneous because they consist of singleton genes (or samples).

The tool for hierarchical clustering of genes can be used for perform a cluster analysis of the genes alone, or a cluster analysis of both the genes and the samples, where the output includes an image plot of the log-ratio values where the genes and the samples are sorted according to their dendrogram order. These two tools will be discussed separately below.

When performing a cluster analysis of genes or samples, BRB-ArrayTools will create a **Cluster viewer** worksheet in the collated project workbook, which will display the dendrogram plot. To continue viewing the sequence of plots, click on the **Next** button at the top of the page. Clicking on the **Next** button may also bring up a prompt for the user to define discrete clusters from the dendrogram by "cutting" the tree, if the analyses selected by the users require the definition of distinct clusters. All plots that are shown in the **Cluster viewer** are also automatically saved as JPEG files in the `Output` folder of the project folder, and may be subsequently edited using other commercially available graphics software.

Several options are available in the clustering procedure for the distance metric and linkage method.

## Distance metric

The options currently available for the distance metric are: “one minus (centered) correlation” and “Euclidean distance”. For clustering of samples only, the “one minus uncentered correlation” is also offered as an additional option.

A Pearson (centered) correlation between two experiments X and Y is defined as

$$\Sigma_{i=1 \text{ to } N} (X_i - X_{avg})(Y_i - Y_{avg}) \,/\, [\, (\Sigma_{i=1 \text{ to } N} (X_i - X_{avg})^2)\, (\Sigma_{i=1 \text{ to } N} (Y_i - Y_{avg})^2)]^{½},$$

where the summation index *i* runs through the N genes in the two experiments, and $X_{avg}$ is the mean over all the genes in experiment X, and $Y_{avg}$ is the mean over all the genes in experiment Y. (The formula is the same for the Pearson correlation between two genes X and Y, except that the summation index *i* would run through the M experiments in the dataset.) Pearson correlation is a commonly used measure of similarity of two columns of numbers, and hence one minus Pearson correlation serves as a distance metric. The two columns used in computing correlation contain the normalized log-ratios or normalized log-intensities of the genes for the two samples being compared.

When the entire set of genes is not used for clustering samples, it may be appropriate to use the “uncentered correlation” metric. The uncentered correlation between two experiments X and Y is defined as

$$\Sigma_{i=1 \text{ to } N} X_i Y_i \,/\, [\, (\Sigma_{i=1 \text{ to } N} X_i^2)\, (\Sigma_{i=1 \text{ to } N} Y_i^2)\, ]^{½},$$

where the summation index *i* runs through the N genes in the two experiments. If the complete set of genes is used and the experiments have been normalized, then the centered correlation between any two experiments is very similar to the uncentered correlation between those two experiments, because the means of the two normalized data columns are very similar.

Euclidean distance is a somewhat different metric. The Euclidean distance between two columns is equivalent to the root-mean-square error between the two columns; it is the square root of the sum of squared differences between the expression levels. The expression profiles of two samples have a large correlation if genes that are highly expressed relative to the mean in one are also highly expressed relative to the mean in the other. The Pearson correlation depends on the patterns of which genes are above and below the mean expression in each profile. Euclidean distance measures absolute differences between expression levels of individual genes rather than on patterns of whether similar genes are up or down regulated relative to each other. The Euclidean distance between two expression profiles is small only if the absolute expression levels of the genes in the two profiles are very similar. The Pearson correlation metric is most commonly used, but it may be worthwhile to examine clustering also with regard to Euclidean distance.

## Linkage

The other option is whether to use the average linkage, complete linkage or single linkage version of the hierarchical clustering algorithm. The distance metric used defines the distance between the expression profiles of two samples. During hierarchical clustering, however, the algorithm must compute the distance between clusters formed at a previous step, or between a singleton and a previously created cluster. With average linkage clustering, the distance between two clusters is taken as the average of the distances between all pairs of elements, one from the first cluster and one from the second. With complete linkage clustering, the distance between two clusters is taken as the maximum distances between an element in the first cluster and an element in the second. With single linkage clustering, the distance between two clusters is taken as the minimum of these distances. Complete linkage clusters tend to be compact, with all members of a cluster relatively equally distant from each other. The results of complete linkage clusters, however, may be heavily dependent upon the elements that were used to start the clusters. Single linkage clusters tend to be “long and narrow” in multi-dimensional space, containing members very distant from each other but closer to some intermediate member. Average linkage clustering is most commonly used because it provides a compromise between the advantages and disadvantages of the other types.

## Cluster analysis of genes (and samples)

The cluster analysis of genes produces a heatmap image (a matrix of plotted varying colors, representing the over- or under-expression of each gene in each sample), where the rows in the image represent the genes, and the columns in the image represent the samples. The heatmap will be shown in an interactive Dynamic Heatmap Viewer tool. You can zoom in to show special regions of the heatmap, and mouse over to show gene and sample names.

There are different metric choices. "1-Correlation" (which centers and scales the genes) or Euclidean distance can be used as the distance matrix. There are also different linkage choices. "Average linkage", "Complete linkage", or "Single linkage" can be used to determine the method of computing the distance between two clusters. Average linkage takes the average of the distances between each possible pair of genes from the two clusters, whereas complete linkage takes the maximum, and single linkage takes the minimum of the distances between each possible pair of genes from the two clusters.

The user has the option to order the samples in the image plot based on a cluster analysis of the samples whose dendrogram-order will determine the plotting order of the samples, or based on a pre-defined experiment descriptor variable whose order will determine the plotting order of the samples. If a cluster analysis of the samples is chosen, then the user has the option to center and scale the genes or not, and a dendrogram will be shown on the top of the heatmap.

In the heatmap, the genes are sorted in dendrogram-order. The gene list and sample list in the heatmap orders can be exported.

## Cluster analysis of samples alone

Clustering samples is frequently performed using the entire set of genes that pass the established filter levels. In certain cases, the user may wish to use a special gene set for clustering and that is an option. There is an option for whether the genes should be median centered in computing the distance between pairs of samples. In most cDNA arrays using an internal reference channel, it is advisable to use the median gene centering option because it reduces the influence of the expression profile of the internal reference sample, which is often not itself of interest.

Another option in the cluster analysis of samples is the ability to compute reproducibility measures. Cluster analysis algorithms always produce clusters, but the clusters are not always reproducible or biologically meaningful. With gene clustering this is less of a problem because we know in advance that gene products are grouped into pathways. It may be a question whether genes in the same cluster are co-regulated, but the existence of biologically meaningful gene clusters is usually not open to question. This is not the case with clusters of samples based on disease tissue from different patients, however. The claim that there are "real" clusters in many cases represents the substantial claim that the disease is molecularly heterogeneous. Such a claim requires more evidence than the fact that the clustering algorithm produced clusters.

Probably the best evidence for reproducibility of clusters of samples is to demonstrate that the clusters that patients tissues are placed in remain the same when the analysis is repeated using RNA independently extracted from each of the same samples. In many cases, however, two experiments based on independently extracted RNA samples for each sample is not available. BRB-ArrayTools provides some alternative, though less compelling, measures of the reproducibility of the sample clusters.

The user may select the cluster reproducibility analysis in the dialog box for clustering samples. The reproducibility analysis is based on perturbing the normalized log-ratios (or normalized log intensities for single label oligonucleotide data) and re-clustering the perturbed data. Indices are computed for each cluster in the original data indicating how much the membership of the cluster changed based on the perturbation. The perturbation and re-clustering process is repeated many times (the number of repetitions is defined by the user) and the output is a summary over the replications of the indices of reproducibility for each cluster. The indices are not computed for all clusters in the dendrogram, but rather for all clusters defined by cutting the original dendrogram at a level defined by the user. A similar level is used for re-clustering the perturbed data in each replication.

Two reproducibility indices are reported. One is called the Reproducibility or R measure for each cluster. The R measure is based on pairs of samples in a cluster of the data. For each such pair of samples, the program computes the proportion of the replications that those two samples remain in the same cluster. That proportion, averaged over replications and over all pairs of samples in the same cluster, is the R measure for that cluster. An R

of 1 means perfect reproducibility of that cluster. An R of 0 means no reproducibility of the cluster.

The other index used is the Discrepancy or D measure. Each cluster of the original data has a "target" cluster in a repetition of the perturbed data. The "target" cluster is that cluster of the perturbed data that contains the largest number of samples included in the original cluster. The program computes the number of discrepancies as the number of samples in the target cluster but not in the original cluster plus the number of samples in the original cluster but not in the target cluster. This number of discrepancies is computed for each original cluster and averaged over the repetitions.

The data is perturbed using Gaussian random noise. The user can specify the variance of the noise if he/she has independent data on the variation over arrays of measurements of independent labelings and hybridization of the same RNA sample. If the user does not specify a value of the variance to be used, the program computes the variance of normalized log-ratios (or normalized log intensities) across all of the arrays for each gene and uses the median variance for the generation of Gaussian perturbations. This estimation is reasonable as long as most genes are not differentially expressed across the arrays. The variance measure used may include some component of biological variation, rather than just experimental variation. The inclusion of biological variability is desirable, although true biological variability would be correlated among sets of genes and the perturbations simulated by the program are generated independently for each gene.

For more information about the cluster reproducibility analysis, see the paper "Methods of assessing reproducibility of clustering patterns observed in analyses of microarray data" by LM McShane, MD Radmacher, B Freidlin, R Yu, MC Li and R Simon in Bioinformatics 18:1462-1469, 2002 also available as a technical report at http://linus.nci.nih.gov/~brb/TechReport.htm

### Interface to Cluster 3.0 and TreeView

BRB-ArrayTools now includes an interface to the Cluster 3.0 and TreeView software originally produced by the Stanford group (http://genome-www.stanford.edu). This interface allows the user to automatically send data from the BRB-ArrayTools project workbook directly into a Cluster and TreeView analysis. For instance, the user may choose to subset and manipulate the data within BRB-ArrayTools, but view the cluster analysis results in TreeView instead of using BRB-ArrayTools' built-in clustering tools.

## Visualizationof samples

Multi-dimensional scalingis a group of methods for representing high-dimensional data graphically in low (usually 2 or 3) dimensions and implemented in Visualization of samples tool. The objective in Visualization of samples is to preserve the pair-wise

similarities or distances between objects in the low-dimensional graphical representation. Visualization of samples analysis is similar to cluster analysis in that one is attempting to examine the relations among samples. But Visualization of samples provides a graphical representation of the pair-wise similarities or distances among samples without forcing the samples into specific clusters.

BRB-ArrayTools provides Visualization of samples analysis of the expression profiles of the samples. We provide a 3-dimensional representation displayed as a rotating cloud of spheres in which each sphere represents a single sample. Samples whose expression profiles are very similar are shown close together. The options for selecting gene sets, and similarity or distance metrics for the multi-dimensional scaling analysis are the same as for the clustering samples tool. In general, the user will use the same selections in both analyses. For more information on the options for the distance metric, please refer to the Distance metric section under the clustering samples tool. When complete data is used (i.e., data with no missing values), then the multi-dimensional scaling analysis using Euclidean distance is equivalent to a priancipal component analysis. When incomplete data is used (i.e., data which has missing values), then the multi-dimensional scaling analysis using Euclidean distance can be thought of as an approximation of a principal components analysis. When Euclidean distance is used on complete data, then BRB-ArrayTools utilizes the first three principal components as the axes for the multi-dimensional scaling representation. The principal components are orthogonal linear combinations of the genes. That is, they represent independent perpendicular dimensions that are rotations of the gene axes (if the thousands of gene axes could be represented). The first principal component is the linear combination of the genes with the largest variance over the samples of all such linear combinations. The second principal component is the linear combination of the genes that is orthogonal (perpendicular) to the first and has the largest variance over the samples of all such orthogonal linear combinations; etc. The first three principal components are usually good choices for multi-dimensional scaling representation, though they are not necessarily the best choice for observing clustered structure.

Options on the multi-dimensional scaling dialog box provide the user control of the labeling of samples in the graphical display.

The 3-D rotating plot contains controls for controlling the axis and speed of rotation. The user may also stop the rotation and use his/her mouse (holding the button down) to brush some of the points to obtain identification of the samples represented by those points.

The Visualization of samples dialog box also provides an option for computing a global statistical significance test of whether the expression profiles are clustered. This global test of clustering is based upon the first three components obtained from a multi-dimensional scaling of the samples, which is equivalent to the first three principal components for complete data (i.e., data with no missing values) when the Euclidean distance metric is used. When computing the multi-dimensional scaling components for input into the global clustering procedure, the computation of the distance matrix is slightly different from the computation used in the rotating scatterplots. This is to ensure

that the resulting multi-dimensional scaling coordinates are as analogous to principal components as possible, since the analogous relationship between multi-dimensional scaling coordinates and principal components exists only when the Euclidean distance is used. If the user chooses the centered correlation metric, then the samples are first centered by their means and standardized by their norms, and then the multi-dimensional scaling components are computed using a Euclidean distance on the resulting centered and scaled sample data. If the user chooses the uncentered correlation metric, then the samples are fist standardized by their norms, and then the multi-dimensional scaling components are computed using a Euclidean distance on the resulting scaled sample data. If the user chooses Euclidean distance as the distance metric, then there is no difference between the multi-dimensional scaling components used in the global test of clustering and that used in the rotating scatterplots.

In v4.2, a new feature has been added that allows the user to specify either a pathway, genelist or a gene symbol for each of the axes. If a pathway/genelist is selected then all the genes in the dataset that match the pathway are used when computing the principle components.

The statistical significance test is based on a null hypothesis that the expression profiles come from the same multivariate Gaussian (normal) distribution. A multivariate Gaussian distribution is a unimodal distribution that represents a single cluster. The global test of clustering can be computed only when the user has at least 30 experiments in his or her dataset. More information about the global tests of clustering is available in "Methods of assessing reproducibility of clustering patterns observed in analyses of microarray data" by LM McShane, MD Radmacher, B Freidlin, R Yu, MC Li and R Simon, Journal of Computational Biology 9:505-511, 2002 and also available as a technical report at http://linus.nci.nih.gov/~brb/TechReport.htm

## Using the classification tools

It is frequently of interest to determine whether samples of different phenotypes or samples collected under different conditions differ with regard to expression profile. For example, one class of samples may consist of breast tumors that contain BRCA1 mutations and the other class may consist of breast tumors that do not contain such mutations (I Hedenfalk, et al., Gene expression profiles of hereditary breast cancer, New England Journal of Medicine 344:549, 2001). Another example is comparing tumors that respond to therapy to those that do not. There are numerous microarray studies that have objectives of this type. This type of problem has been called "class prediction" in distinction to "class discovery" because the classes, or phenotypes, of the samples to be compared are known in advance of the expression profiling.

Cluster analysis is not usually the most effective tool for addressing class prediction problems. Clustering of samples is usually based on the entire set of genes represented on the experiment. Since the classes to be distinguished may differ only with regard to a relatively small subset of these genes, these differences may be dominated by variations among the thousands of other genes used in the clustering distance metric. Whereas

perusal of the image plot associated with clustering the samples and the genes may reveal some genes that appear to distinguish the classes, the visual approach is error prone without statistical confirmation that the clusters observed do not represent patterns obtained by chance from screening thousands of genes for those that sort the samples.

BRB-ArrayTools contains two powerful tools for comparing expression profiles among pre-defined classes. They are found under the **Class comparison** and **Class prediction** menu items. Both tools presume that the data consists of experiments of different samples representative of the classes. The experiments should represent replication at the highest level, incorporating biological variability. That is, if we are comparing expression profiles of BRCA1 mutated breast tumors to non-BRCA1 mutated breast tumors, we need samples from breast tumors of different patients in both categories. It is not sufficient to compare replicate experiments of one RNA sample from a BRCA1 mutated tumor to one RNA sample from a non-BRCA1 mutated breast tumor. Comparing two RNA samples, regardless of how many experiments have been run with those two samples, cannot support conclusions about the influence of BRCA1 mutation on expression profiles. If cDNA arrays are used, the two class prediction analyses in BRB-ArrayTools assume that a common internal reference sample has been used in all experiments, or that a patient-specific normal tissue reference is used for the experiments.

## Class comparison analyses

## Class comparison between groups of arrays

The **Class Comparison Between Groups of Arrays Tool** is used for comparing 2 or more pre-defined classes. The classes to be compared are defined by a column of the experiment design worksheet. The codes used in a column can be any set of numerical, character or character string codes. If an entry for a particular experiment is left blank in defining the column, that experiment will be omitted from the class comparison analysis. The ability of the user to define columns in the experiment design worksheet to specify class comparison analyses enables the user to import his/her complete set of experiments into BRB-ArrayTools once and to conduct a variety of different comparisons guided by different columns.

Two options presented for the Class Comparison Between Groups of Arrays Tool are important for some situations: the ability to pair samples, and the ability to average over replicate experiments. If two classes are compared and the experiments are paired, then the paired t-test option should be selected. For example, if experiments have been prepared for the primary tumor and metastatic tumors of each patient, then the paired t-test option is appropriate and may improve the statistical power of the analysis. If multiple technical replicates have been performed for some RNA samples, then the analysis must either be based on selection of a single replicate for each RNA sample or the averaging option should be used. The test is based on comparing the differences in mean log-ratios (or log-intensities) between classes relative to the variation expected in the mean differences. The variation is computed assuming that all the samples are independent. If there are multiple experiments for the sample RNA sample, then the within-class variation will under-estimate the inter-class variation in means to be expected. The user should either omit technically inferior experiments, arbitrarily choose between technically satisfactory experiments that are tightly correlated (as determined using the Scatterplot tool), or utilize the averaging option. A combination of these approaches may also be used. If the averaging option is selected, the user must specify a column of the experiment design worksheet that provides RNA sample identifiers so that the tool can identify when there are multiple experiments for the same RNA sample. When analyzing paired sample, samples with replicated pairing id and classification labels will automatically be averaged. For further information, refer to the following section entitled Specifying replicate experiments and paired samples. The minimum number of arrays required to run the Class Comparison tool is atleast two arrays per class.

The Class Comparison Between Groups of Arrays Tool computes a t-test or F-test separately for each gene using the normalized log-ratios for cDNA arrays and the normalized log-intensities for one color oligonucleotide arrays. The F-test is a generalization of the two-sample t-test for comparing values among groups. The user also has the option of using the random variance version of the t-test or F-test. This is generally advisable, particularly when the number of samples per class is small. The random variance tests are discussed in more detail below. They provide for sharing information among genes of the within-class variance in log-ratios or log signals. The class comparison tool computes the number of genes that are differentially expressed

among the classes at the statistical significance level selected in the F-test menu and creates a gene list containing information about the significant genes.

Several other important statistics are also computed. The tool performs random permutations of the class labels (i.e., which experiments correspond to which classes). For each random permutation, all of the t-tests, F-tests or random variance t-tests and F-tests are re-computed for each gene. The Class Comparison Between Groups of Arrays Tool computes the proportion of the random permutations that gave as many genes significant at the level selected by the user as were found in comparing the true class labels. That p value provides a global test of whether the expression profiles in the classes are different. This test is also robust; although it uses the number of genes significant by the F-test as a statistic, it generates the permutation p-value of that statistic. In comparing classes, it is statistically easier to determine reliably whether expression profiles for pre-defined classes are different than to reliably determine exactly which genes are differentially expressed among the classes. The latter problem more directly confronts the multiple comparison problem.

For example, suppose that we select a significance level of 0.001 for including genes in the gene list. If there are 8000 genes on the experiment, then we expect by chance that 8 genes on the gene list will be false positives. If we obtain 24 genes on the gene list, then about one third of them are false positives. If we obtain 80 genes on the gene list, then about one tenth of them are false positives. Traditionally multiple comparison corrections used in statistical analyses are much more stringent, usually requiring that the chance of any false positives be very small. For most microarray analyses, such conservatism is not appropriate. However, gene lists will not be a useful basis for further experimentation if they are heavily populated by false positives. For example, if we select a significance level of 0.01 instead of 0.001, then we expect by chance that 80 genes on the gene list will be false positives. Having 80 false positives makes interpretation or planning confirmatory experiments very problematic. So using a relatively stringent 0.001 is usually appropriate. With that stringency, however, there may be a substantial chance of false negatives; that is, some genes that are differentially expressed among the classes will not be found significant at the 0.001 level. The false negative rate will depend on the sample size, the within-class variation among samples and the average fold difference between classes for the gene. Hence, although the statistical power for detecting individual genes is limited by the stringency needed to control the false positive rate, the global test of whether the classes differ with regard to expression profiles will have better power for establishing that the expression profiles are different. Because a single global test is performed, obtaining a p value of less than 0.05 is sufficient to establish that the expression profiles are different. Multiple comparison stringency needed for inferences about individual genes is not needed for the global test.

The Class Comparison Tool also provides univariate or multivariate permutation tests for the significance of individual genes. The univariate permutation tests are computed separately for each gene. The proportion of the permutations of the class label giving a t-test or F-test p value as small as obtained with the true class labels is the univariate permutation p value for that gene. Additionally beginning with BRB-ArrayToolsv3.4.0, if

the user requests that the gene list consist of those genes significant at a specified threshold p value, then the multivariate permutation tests are not performed. The output gene list table is ordered by univariate p value with the most significant genes listed first. One of the columns contains an estimate of the false discovery rates. The false discovery rate associated with a row of the table is an estimate of the proportion of the genes with univariate p values less than or equal to the one in that row that represent false positives. The method of Benjami and Hochberg is used for this estimation. This method is less accurate than the method used in the multivariate permutation test method, but is easily computed and has been widely used. For the i'th gene in the table, the estimated false discovery rate is $m \times p_i / i$ where $p_i$ is the univariate p value for the i'th most significant gene and m is the total number of genes tested (after filtering).
Benjamini Y and Hochberg Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society B 57: 289-300, 1995.

When the univariate significance level option is selected, the user will be given the option of computing univariate p values based on permutation t-statistics (or univariate F statistics if there are more than 2 groups) in addition to the t-tests (or F-tests). We have made the default not to report the univariate permutation significance levels for the following reason. When the number of samples is large, the permutation test will generally agree with the t/F test. When the number of samples is small, the permutation tests lack power for establishing statistical significance at stringent threshold p values. Additionally, the output will always contain a permutation based global test of the null hypothesis that there is no relationship between expression profile and class label. Suppose, for example, the user has requested that the gene list consist of genes with univariate significance levels (t/F tests) of 0.001 or less and suppose that n such genes are obtained. The global test will generate random permutations of the class labels and compute the t/F tests for all genes using the randomly shuffled class labels to compare the classes. The output will contain the proportion of randomly shuffled permutations for which there were at least n genes found that were significant at the 0.001 level or less.

The multivariate permutation tests are much more effective than the univariate permutation tests when the number of samples per class are small because there may not be enough distinct permutations to give a permutational p value below a stringent level such as 0.001. The multivariate permutation tests are described in more detail in a later section. They provide the ability to control the number of genes in the "discovery list" that are false discoveries (i.e. false positives) and the ability to control the proportion of false discoveries in the discovery list. For example, the user can specify that he/she wants 90% confidence that the discovery list contains no more than 10% false discoveries.

The multivariate permutation tests are similar in spirit to the Statistical Analysis of Microarrays (SAM) method, although they provide tighter probabilistic control on the number and proportion of false discoveries (Tucher et al. 2001 in reference list). SAM analysis is provided as a separate tool.

An option in the Class Comparison Between Groups of Arrays Tool allows the user to control for a potentially confounding factor while comparing classes. For example, you could compare expression profiles of tissue from two types of tumors while controlling for the gender of the patients from which the tissue was taken. You could also control for technical factors such as print set of the arrays. The analysis performed is an analysis of variance for a randomized block design. Two linear models are fit to the expression data for each gene. The full model includes class variable and the block variable, and the reduced model includes only the block variable. Likelihood ratio test statistics are used to investigate the significance of the difference between the classes. The random variance option is also available for this analysis of variance and multivariate permutation tests are applied to the resulting significance levels.

New to v3.8, is an option to restrict the significant genelist by a fold change criteria. This is available under the “options” button on the dialog. The HTML output has been enhanced in this release to include a heat map of the significant genes. This heat map is generated after clustering the significant genes and samples. The ‘grey’ color scale is used for single channel data and the default red-green color scale is used for dual channel data. The parameter to control the level of truncation can be modified in the “preference” option. Under the “ArrayTools” pull down menu there an option called “utility”->”Preference” and you can change the parameter “Heatmapthreshold”. Also included in this release is an interactive plot namely ‘volcano’ plot for two classes or ‘parallel coordinate plot’ for more than two classes. Each point in the volcano plot for example represents a gene from the list of significant genes. By clicking on any point in the plot ,the corresponding gene symbol will appear.New to v4.1 , a pair-wise comparison on a class variable with more than 2 class levels will be automatically tested. The test is based on simple T-test. Users can change the default P-value threshold from the 'Options' dialog. On the HTML output, a new column 'Pairwise significant' will be included in the table of "Genes which are differentially expressed among classes". For example, (1,3) means class 1 and class 3 are different for that specific gene at the specified P-value threshold.

## Class comparison between red and green channels

In v3.6, the Class comparison between red and green channels has been replaced by an enhanced ANOVA plug-in of log intensities. The plug-in is used for finding genes differentially expressed between two classes for two-color arrays without a common reference sample. It can also be used to compare samples of one class with the reference samples in the common reference design. For additional details on the plug-in you can refer to ANOVA on log-intensities

## Gene Set Comparison Tool

The gene set comparison tool analyzes pre-defined gene sets for differential expression among pre-defined classes. The user chooses which pre-defined gene sets to analyze, which may be defined by user own genelists or based on Gene Ontology categories, BioCarta pathways, KEGG pathways, protein domains, transcription factor targets, microRNA targets, and the Broad Institute's Molecular Signature DataBase (MsigDB) gene set collections. We will describe the Gene Set Comparison tool in the context of evaluating Gene Ontology gene sets.

The evaluation of which GO categories are differentially expressed among phenotype classes is performed using a functional class scoring analysis as described by Pavlidis et al. (2004). Functional class scoring is a more powerful method of identifying differentially expressed gene classes than the more common over-representation analysis or annotation of gene lists based on individually analyzed genes. It indicates which gene sets contain more differentially expressed genes than would be expected by chance.

First a p-value is computed for each gene in a GO category. Then the set of p-values for a GO category is summarized by the LS and KS summary statistics. For a set of N genes, the LS statistic

$$LS = \sum_{i=1}^{N}(-\log(p_i))/N$$

is defined as the mean negative natural logarithm of the p-values of the appropriate single gene univariate test. The KS statistic

$$KS = \max_{i=1}^{N} (i/N-p_i), \qquad p_1 \leq p_2 \leq \ldots \leq p_N$$

is defined as the maximum difference between i/N and $p_i$, where $p_i$ is the $i^{th}$ smallest p-value of the univariate test. This is the Kolmogorov-Smirnov statistic for testing if the p-values are of a uniform distribution.

The statistical significance of a GO category containing N genes is evaluated by computing the empirical distribution of these summary statistics in random samples of N genes. The significance values are based on testing the null hypothesis that the list of genes that belong to a given GO category is a random selection from the project gene list, against the alternative hypothesis that it contains more genes differentially expressed between the classes being compared. WARNING: Only use the gene set comparison tool with a gene list that has been subjected to non-specific filtering. Do not use this tool with a list of differentially expressed genes identified by class comparison analysis in BRB-Array Tools as the statistics are invalid". The re-sampling p-value is not based on normal distribution assumptions. Rather, it is based on a re-sampling procedure. For a given GO category, the re-sampling procedure randomly selects N genes from the list of genes that are selected for the analysis. Here N is the number of genes in the project gene list that belong to the GO category. For each selection, the associated LS and KS statistics are computed. The selection process is repeated 100,000 times to obtain a distribution of these statistics. Then, for each GO category, the LS (KS) permutation p-value is defined

as the proportion of selections for which the LS (KS) statistic is larger then the LS (KS) statistics computed for the GO category with original gene list.

The tests are applied separately to each GO category. A GO category is selected if its corresponding LS or KS re-sampling p-value is below the threshold specified by the user (default is 0.005). Some approximations are used to speed up computations.

Results are provided as a table of selected GO categories that are ordered by the p-value of the LS test (smallest first). For each GO category, the table lists the unique identifier, the number of genes in the project gene list that belong to the GO category, and the LS and KS p-values. For each class, the geometric mean of gene expression (ratios) is provided. If only two classes are considered, the fold difference of these geometric means is also listed.

Another table lists all genes that are found in the selected GO categories. They are ordered by the p-value associated with the GO category. If a gene belongs to several GO categories, it will be listed several times, once for each GO category that contains the gene.

Since BRB-ArrayTools v3.7 a new gene set analysis method, Efron-Tibshirani's Gene Set Analysis (GSA, Efron & Tibshirani 2007), is implemented to use "maxmean" statistics improved upon the original Gene Set Enrichment Analysis (GSEA) procedure of Subramanian et al. (2005) for assessing signficance of pre-defined gene-sets. GSA uses the maxmean statistic: this is the man of the positive or negative part of gene scores di in the gene set, whichever is large in absolute value. Specifically speaking, take all of the gene scores di in the gene set, and set all of the negative ones to zero. The take the average of the positive scores and the zeros, giving a positive part average avpos. Do the same for the negative side, setting the positive scores to zero, giving the negative part average avneg. Finally the score for the gene set is avpos if |avpos| > |avneg|, and otherwise it is avneg. Efron and Tibshirani shows that this is often more powerful than the modified Kolmogorov-Smirnov statistic used in GSEA. The R-package GSA is automatically imported by ArrayTools and used for calculation.

The Gene Set Comparison Tool can be used when user-defined gene lists are investigated rather than GO categories. The tool investigates the user-defined gene lists and selects those that have more genes differentially expressed among the classes than expected by chance.

With pathway comparisons the genes are grouped by KEGG or BioCarta pathways, rather than by GO categories. "BioCarta" is a trademark of BioCarta, Incorporated, and the pathways included in BRB-ArrayTools provide displays of gene interactions within pathways for human and mouse cellular processes. "KEGG" (Kyoto Encyclopedia of Genes and Genomes) is an original database product developed by Kinoru Kanehisa and associates, and the pathways included in BRB-ArrayTools provide displays of

interactions between enzymes and intermediary compounds within metabolic and regulatory pathways. The KEGG and BioCarta pathways used within BRB-ArrayTools are obtained from the Cancer Genome Anatomy Project (http://cgap.nci.nih.gov/Pathways), and are available to non-commerical users.A new feature added to v3.4 allows the user the option to download and use the Broad/MIT pathways and signatures (http://www.broad.mit.edu/gsea/msigdb/msigdb_index.html). Once the user agrees to the terms and conditions from the Broad Institute, the relevant files get downloaded and parsed. From v3.7, ArrayTools extends to incorporate the complete gene set collections from the Broad Institute's Molecular Signature DataBase (MsigDB). The database is organized into 4 broad collections: Positional gene sets, Curated gene sets, Motif gene sets, Computed gene sets. For descriptions of the collections, see the MSigDB release notes. The previous Broad/MIT gene sets available between release v3.4 and v3.6 are now part of the Curated gene sets collection in release v3.7. ArrayTools provides a convenient way for user to register and login on the Broad Institute Website and downloads the gene set collections for use within gene set comparison tool. Also, now the Broad/MIT gene sets are available for both the Human and Rat species. However, only the curated and motif gene sets are available for Rat species.

From release v3.6, three additional gene sets have been included, namely the transcription factor targets, microRNA targets, and protein domains.
For gene sets of transcription factor targets, all genes in each gene set are either predicted or experimentally verified to be targets of the same transcription factor. Separate sets of human and mouse genes are provided. The Predicted targets were obtained using the web-based software MATCH [1] (version 10.0) to search the upstream sequences of genes (~1500bp). The sequences were obtained from the ENSEMBLE SANGER (version 40) database. The search utilized transcription factor binding weight matrices obtained from the TRANSFAC [2] (version 9.2) database and the MATCH cut-offs to minimize the number of false positive targets. With this approach, each set contains genes that are predicted to be potential targets of the same transcription factor. Additionally, also included are separate sets of genes that have been experimentally verified as targets of the same transcription factor. We used transcription factor binding curation information in the Transcriptional Regulatory Element Database (TRED) [3] to eliminate targets without any experimental verification.

Gene sets of predicted mircoRNA targets are also included. The predicted miRNA target genes information in the miRBase Targets (version 4) database [7] to group genes into sets. Genes in each set are predicted to be potential targets of the same miRNA. The prediction steps include first detecting potential binding sites with a large degree of complementarity's of the miRNA, followed by filtering out those sites that do not appear to be conserved in multiple species. Separate sets of human and mouse genes are provided.

In v3.6-Beta3, an additional family of gene sets was added from the PFAM and SMART Protein Domain. These are gene sets that contain genes whose protein products share a common domain. We used Pfam [4] and SMART [5] protein domain links in the Swiss-

Prot and TrEMBL [6] (version 9.3) databases to group genes into sets. Proteins encoded by genes in each set contain the same domain. Pfam and SMART are high quality manually curated protein domain databases. Separate sets of human and mouse genes are provided. The reference can be found on http://linus.nci.nih.gov/techreport/Xu-GeneSetToolkit.pdf

In the previous v3.8 release, an option was added to handle redundant probe sets that correspond to the same gene. However, this option is now available in the Filtering and gene sub-setting tool. Additionally, the HTML output now contains a heatmap for each significant gene set. The heat map is in a grey scale for single channel data and red-green scale for dual channel data. The parameter to control the level of truncation can be modified in the "preference" option. Under the "ArrayTools" pull down menu there an option called "utility"->"Preference" and you can change the parameter "`Heatmapthreshold`".

For BRB-ArrayTools v4.1, the gene set analysis tool has been extended to provide an option for interaction analysis. Traditionally, gene set analysis finds sets of genes that are differentially expressed among defined classes of samples. For example, the classes might represent samples from patients who respond to a treatment versus samples for patients who do not respond. Interaction analysis finds gene sets for which the differential expression among classes is different for two pre-defined groups of samples. The groups might represent patients with different stages of disease. Testing for interactions is statistically preferable to the usual practice of comparing the gene sets significant for one group to the gene sets significant for the other group.

The general testing procedure is as follows. We start by computing a nominal statistical significance value $p_i$ measuring differential expression between classes for each gene i using a univariate t test or random variance t test. These $p_i$ values are computed separately for the two groups. We would call them $p_i$ values for group 1 and $q_i$ values for group 2. Then for each gene set k we compute a summary statistic $P_k$ of the $p_i$ values for i in k for samples in group 1 and a summary statistic $Q_k$ of the $q_i$ values for i in k for samples in group 2.

We then test the hypothesis that the two groups are equivalent with regard to inter-class variation in gene expression for the genes in set k. We summarize the difference between groups using a test statistic $P_k$-$Q_k$.

To obtain the null distribution of the test statistic $P_k$-$Q_k$ we randomly permute the group labels, keeping the class labels unchanged and re-calculating P[k], Q[k] and the test statistic for the permuted data. We do this thousands of times and thereby tabulate the null permutation distribution of $P_k$-$Q_k$. We calculate a two-sided significance level for each gene set k using these null permutation distributions.

Two kinds of summary statistics have been implemented: LS test and GSA test. For the LS test, the summary statistic $P_k$ is defined as the sum of the -log($p_i$) for i in k. For the

GSA test, the summary statistic is the maxmean statistic as defined in Efron & Tibshirani (2007).

References:

1. Gobling et al. (2001) MATCH™-a tool for searching transcription factor binding sites in DNA sequences. Proceedings of the German conference on Bioinformatics GCB'01, pp.158-161.

2. Matys et al. (2003) TRANSFAC: transcriptional regulation, from patterns to profiles. Nucleic Acids Research 31: 374-378.

3. Zhao et al. (2005) TRED: a Transcriptional Regulatory Element Database and a platform for in silico gene regulation studies. Nucleic Acids Research 33: D103-107.

4. Bateman et al. (2000) The Pfam protein families database. Nucleic Acids Research 28: 263-266.

5. Letunic et al. (2002) Recent improvements to the SMART domain-based sequence annotation resource. Nucleic Acids Research 30: 242-244.

6. Boeckmann et al. (2003) The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. Nucleic Acids Research 31: 365-370.

7. Griffiths-Jones et al. (2006) miRBase: microRNA sequences, targets and gene nomenclature. Nucleic Acids.

10. Subramanian et al. (2005) Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. Proc. Natl. Acad. Sci. USA 102, 15545-15550.

11. Efron B, Tishirani R. (2007) On testing the significance of sets of genes. Annals of Applied Statistics, Vol. 1, No.1, 107-129.

## Significance Analysis of Microarrays (SAM)

The **Significance Analysis of Microarrays (SAM) Tool** is based upon a popular algorithm developed by Tucher et al. (reference below) for identifying significant genes in microarray data (www-stat.stanford.edu/~tibs/SAM). The SAM algorithm is one method of controlling the False Discovery Rate (FDR), which is defined in SAM as the median number of false positive genes divided by the number of significant genes. The SAM algorithm is an alternative to the multivariate permuation test provided in several BRB-ArrayTools class comparison tools. The multivariate permutation test has a stronger statistical basis than SAM but we include SAM because some users are more familiar with it.

First we compute for each gene a modified F-statistic (or t-statistic for two-class data) in which a "fudge factor for standard deviation" is included in the denominator to stabilize the gene specific standard deviation estimates. We order these F-statistics by sorting them from smallest to largest ($F_{(1)}$, $F_{(2)}$, …, $F_{(i)}$, …, $F_{(n)}$), where n is the number of genes. Then we permute the class labels and re-compute a set of ordered F-statistics for each permutation. The expected ordered statistics are estimated as the average of the ordered statistics over the set of permutations. A cutpoint is then defined as $F_{(i^*)}(\Delta)$, where $i^*(\Delta)$ is the first index $i$ in which the actual ordered F-statistic is larger than the expected ordered F-statistic by a $\Delta$-threshold value, and is a function of this $\Delta$. Genes which have an F-statistic larger than this cutpoint are considered to be "significant". For random permutations, any "significant" genes are presumed to be false positives, and a median number of false positive genes can be computed over the set of permutations. The median number of false positive genes is then multiplied by a shrinkage factor $\pi$, which represents the proportion of true null genes in the dataset, and is computed as the number of actual F-statistics which fall within the interquartile range of the set of F-statistics computed for all permutations and all genes, divided by the quantity of .5 times the number of genes. If this $\pi$ factor is greater than 1, then a $\pi$ factor of 1 is used instead. The median number of false positive genes, multiplied by $\pi$ and divided by the number of "significant" genes, yields the FDR for a given $\Delta$ value.

In the SAM tool implemented in BRB-ArrayTools, the user specifies the desired False Discovery Rate (FDR) and number of permutations to run, and BRB-ArrayTools will automatically search through a range of $\Delta$ values to find an optimal $\Delta^*$ that yields the FDR specified by the user. The "fudge factor for standard deviation" is obtained and fixed a priori by searching through a range of values, and finding a value which minimizes the coefficient of variation of the median absolute deviation of the modified F-statistics. More details about the algorithm may be found in Tusher, et al. (VG Tusher, R Tibshirani, G Chu, Significance analysis of microarrays applied to the ionizing radiation response, Proceedings of the National Academy of Science USA 98:9:5116-5121, 2001).

The output of the SAM analysis contains the computed "fudge factor" and the optimal $\Delta^*$ which yields the desired FDR, a list of the significant genes (defined as those genes for which the actual ordered F-statistic is larger than the expected ordered F-statistic by the $\Delta^*$ threshold), and a plot of the actual vs. expected ordered F-statistics. Also, a barplot is given to compare the chromosomal distribution of the selected genelist to that of the overall geneset, and a table of observed vs. expected counts is given for Gene Ontology classes which are over-represented in the selected genelist. In BRB-ArrayTools v3.3 we have re coded the SAM algorithm in Fortran and now it is almost 8 times faster to run the SAM analysis.

# Class prediction analyses

## Class prediction

The **Class Prediction Tool** is similar to the Class Comparison Between Groups of Arrays Tool in that the classes must be pre-defined. The user specifies the classes by identifying a column of the experiment description worksheet. Blanks labels in the column indicate that the corresponding experiment should be omitted from the analysis. Several of the class prediction methods are only applicable when there are two classes. Replicate experiments of the same RNA sample may be averaged, by entering a sample id column in the experiment design worksheet. When analyzing paired samples, samples with replicated pairing id and classification labels will automatically be averaged. For further information, refer to the following section entitled Specifying replicate experiments and paired samples.

For dual-label experiments, the class prediction methods are based on normalized log-ratios relative to a common reference RNA sample. For Affymetrix GeneChip™ arrays the methods are applied to normalized log signal values. To reduce the dominating effect that genes with overall high intensity might have with some class prediction methods, the single-channel log-signal values are median-centered on a gene-by-gene basis for class prediction. This is not done for class comparison analyses because the results there are un-affected by median centering the genes.

## Gene selection for inclusion in the predictors

Generally, the user specifies a significance level to be used for determining the genes that will be included in the predictors; genes that are differentially expressed between the classes at a univariate parametric significance level less than the specified threshold are included in the predictor. It doesn't matter whether the specified significance level is small enough to exclude enough false discoveries, because class prediction is not really about discovering differentially expressed genes. In some problems better prediction can be achieved by being more liberal about the gene sets used as features. The predictors may be more biologically interpretable and clinically applicable, however, if fewer genes are included. As noted below in the section on cross-validation, gene selection is repeated for each training set created in the cross-validation process. That is for the purpose of providing an unbiased estimate of prediction error. The final model and gene set for use with future data is the one resulting from application of the gene selection and classifier fitting to the full dataset.

Because the user generally doesn't know whether better prediction would be obtained from a more stringent or less stringent threshold significance level for gene selection, an option is provided to develop the classifiers over a grid of significance levels for gene selection. The cross-validated prediction error is computed for all significance levels in the grid and for all classifier functions selected. The significance level threshold used for gene selection can be viewed as a "tuning parameter" for the development of a classifier. Selecting the value of a tuning parameter to give smallest cross-validated prediction error should be viewed as part of the classifier determination algorithm. Providing the cross-validated prediction error for a range of values of the tuning parameter and inviting the user to select the tuning parameter corresponding to the minimum prediction error is

somewhat dangerous however. The minimum cross-validated prediction error is a biased estimate of the prediction error to be expected for future data. Therefore, BRB-ArrayTools provides a completely cross-validated estimate of the prediction error to be expected with independent data based on the optimal selection of the tuning parameter. This completely cross-validated estimate is obtained by doing a doubly nested cross-validation; the outer loop for estimating the prediction error for a test set consisting of one or more samples omitted from the training data, and the inner loop used for optimizing the tuning parameter. This is quite computationally intensive, however.

The gene selection methods described above ranks genes based on the extent to which they are individually differentially expressed among the classes. Some of the classifier functions then provide a multivariate modeling of the selected genes, but the genes to be incorporated in the model are selected based on univariate discrimination ability. BRB-ArrayTools provides, however, one other option for selecting genes, namely the greedy-pairs method described by Bo and Jonassen (Genome Biology 3(4):research0017.1-0017.11, 2002). The greedy-pairs approach starts with ranking all genes based on their individual t-scores on the training set. The procedure selects the best ranked gene $g_i$ and finds the one other gene $g_j$ that together with $g_i$ provides the best discrimination using as a measure the distance between centroids of the two classes with regard to the two genes when projected to the diagonal linear discriminant axis. These two selected genes are then removed from the gene set and the procedure is repeated on the remaining set until the specified number of genes have been selected. This method attempts to select pairs of genes that work well together to discriminate the classes. It is done in a computationally efficient manner. The user does have to specify the number of total genes to be selected (twice the number of pairs to be selected). As for all BRB-ArrayTools algorithms for gene selection, the process is repeated for all training sets created during the cross-validation process.

The Support Vector Machine Recursive Feature Elimination (SVM RFE) method uses an SVM classifier trained on the data to rank genes according to their contribution to the prediction performance. The SVM algorithm uses a weighted linear combination of the gene expressions as a discriminator between the two classes. This linear combination is selected to maximize the margin, or the distance between the worst classified samples and the discriminant plane.

Initially expression of all the genes is used to train the SVM algorithm. The SVM RFE algorithm removes genes that have a low absolute value of weight in the linear combination and a new SVM classifier is developed using the remaining genes. After finding a new linear discriminant, the genes that have the lowest absolute weights in the new discriminant are removed and a new SVM classifier is developed. Etc. The process continues iteratively until the desired number of genes is left.

Although this feature selection method uses SVM classifiers, the features can be used with any of the classification methods. If one is performing LOOCV, the feature selection is performed independently for each leave-one-out training set; and similarly for other cross-validation methods

The Class Prediction Tool creates a multivariate predictor for determining to which of the two classes a given sample belongs. Several multivariate classification methods are available, including the Compound Covariate Predictor, Diagonal Linear Discriminant Analysis, Nearest Neighbor Predictor, Nearest Centroid Predictor, and Support Vector Machine Predictor.

### Compound covariate predictor

The Compound Covariate Predictor is a weighted linear combination of log-ratios (or log intensities for single-channel experiments) for genes that are univariately significant at the specified level. By specifying a more stringent significance level, fewer genes are included in the multivariate predictor. Genes in which larger values of the log-ratio pre-dispose to class 2 rather than class 1 have weights of one sign, whereas genes in which larger values of the log-ratios pre-dispose to class 1 rather than class 2 have weights of the opposite sign. The univariate t-statistics for comparing the classes are used as the weights. Detailed information about the Compound Covariate Predictor is available in the Hedenfalk reference given above or in "A paradigm for class prediction using gene expression profiles" by MD Radmacher, LM McShane and R Simon, A paradigm for class prediction using gene expression profiles. Journal of Computational Biology 9:505-511, 2002, also available in Technical Report 01, 2001, Biometric Research Branch, National Cancer Institute, http://linus.nci.nih.gov/~brb/TechReport.htm).

### Diagonal linear discriminant analysis

The Diagonal Linear Discriminant Analysis is similar to the Compound Covariate Predictor, but not identical. It is a version of linear discriminant analysis that ignores correlations among the genes in order to avoid over-fitting the data. Many complex methods have too many parameters for the amount of data available. Consequently they appear to fit the training data used to estimate the parameters of the model, but they have poor prediction performance for independent data. The study by Dudoit et al. (S Dudoit, J Fridlyand, TP Speed; Comparison of discrimination methods for the classification of tumors using gene expression data, Journal of the American Statistical Association 97:77-87, 2002) found that diagonal linear discriminant analysis performed as well as much more complicated methods on a range of microarray data seta.

### Nearest neighbor predictor

The Nearest Neighbor Predictor is based on determining which expression profile in the training set is most similar to the expression profile of the specimen whose class is to be predicted. The expression profile is a vector of log-ratios or log-intensities for the genes selected for inclusion in the multivariate predictor. Euclidean distance is used as the distance metric for the Nearest Neighbor Predictor. Once the nearest neighbor in the

training set of the test specimen is determined, the class of that nearest neighbor is taken as the prediction of the class of the test specimen. k-Nearest Neighbor Prediction is similar. For example with the 3-Nearest Neighbor algorithm, the expression profile of the test specimen is compared to the expression profiles of all of the specimens in the training set and the 3 specimens in the training set most similar to the expression profile of the test specimen are determined. The distance metric is again Euclidean distance with regard to the genes that are univariately significantly differentially expressed between the two classes at the threshold significance level specified. Once the 3 nearest specimens are identified, their classes vote and the majority class among the 3 is the class predicted for the test specimen.

## Nearest centroid predictor

The Class Prediction Tool also offers Nearest Centroid Prediction. In the training set there are samples belonging to class 1 and to class 2. The centroid of each class is determined. The centroid of class 1, for example, is a vector containing the means of the log-ratios (or log intensities for single label data) of the training samples in class 1. There is a component of the centroid vector for each gene represented in the multivariate predictor; that is, for each gene that is univariately significantly differentially expressed between the two classes at the threshold significance level specified. The distance of the expression profile for the test sample to each of the two centroids is measured and the test sample is predicted to belong to the class corresponding to the nearest centroid.

## Support vector machine predictor

The Class Prediction Tool also offers Support Vector Machine Prediction. A support vector machine (SVM) is a class prediction algorithm that has appeared effective in other contexts and is currently of great interest to the machine learning community. SVMs were developed by V. Vapnik (The Nature of Statistical Learning. Springer-Verlag, New York, 1995). We have implemented SVMs with linear kernel functions as our experience has been that more complex SVMs perform less well for this application. The SVM predictor is a linear function of the log-ratios or the log-intensities that best separates the data subject to penalty costs on the number of specimens misclassified. In the options dialog, the user can alter the default penalty cost or the cost of misclassifying a specimen of class 1 relative to misclassifying a specimen of class 2. We use the LIBSVM implementation of Chang and Lin.

### Bayesian compound covariate predictor

The Bayesian compound covariate predictor is based on the method described by G Wright, B Tan, A Rosenwald, EH Hurt, A Wiestner and LM Staudt (A gene expression-based method to diagnose clinically distinct subgroups of diffuse large B cell lymphoma, PNAS 100:9991-9996, 2003). After selecting the differentially expressed genes for distinguishing two classes in a cross-validated training set, the compound covariate is

computed which is the weighted average of the log expression values of the selected genes, with the weights being the t statistics of differential expression in that training set. That compound covariate can be computed for all of the samples in the cross-validated training set as well as for the sample(s) omitted from the training set. The values of the compound covariate scores of samples in each class in the training set are considered to have a Gaussian distribution. These are univariate Gaussian distributions. The means of the Gaussian distributions differ among the classes for the samples in the training set but the variances are assumed to be equal and a pooled estimate is used. With this model, using a Bayesian context, the posterior probability that the omitted sample belongs to class 1 can be written

P(class 1 | $\underline{x}$) = P($\underline{x}$ | class 1) * Prior(class 1) / { P($\underline{x}$ | class 1) * Prior(class 1) + P($\underline{x}$ | class 2) * Prior(class 2)}.

In this formula $\underline{x}$ denotes the vector of log expression values for the omitted sample with regard to the genes selected for use in the classifier developed in the training set with that observation omitted. Because of the assumption of Gaussian distributions, P($\underline{x}$ | class 1) is the univariate Guassian density function for the compound covariate score for class 1 in the training set and P(x | class 2) is the Gaussian density function for the compound covariate score for class 2 in the training set. Prior(class 1) and Prior(class 2) are the assumed prior probabilities for the two classes. The non-Bayesian compound covariate predictor is implicitly based on the assumption that the prior probabilities are each ½ . So if you select that option, the Bayesian compound covariate predictor with give the same predictions as the standard CCP. We have included that option so that users can see which predictions are clear-cut and which are less clear-cut. That is, if the posterior probability that an omitted observation is in class 1 is close to 1 or to 0, then that prediction is clear-cut. If the posterior probability is close to 0.5, then it's a toss-up. We also provide an option of using prior probabilities that correspond to the relative frequencies of the classes in the cross-validated training sets.

After computing the posterior probability of the class of an omitted observation we assign the observation to the class with the larger posterior probability. In the next release we will provide an option of providing no prediction if the posterior probabilities are within a specified threshold of 0.5.

## Cross-validation and permutation p-value

For all of the class prediction methods requested, the Class Prediction Tool provides an estimate of how accurately the classes can be predicted by this multivariate class predictor. The cross-validated misclassification rate is computed and the results are reported. Several options are provided for cross-validation. Leave-one-out cross-validation (LOOCV) is often used. With LOOCV, the cross-validation process omits one sample at a time. For each sample omitted, the entire analysis is repeated from scratch, including the determination of which genes are univariately significant on the reduced training sample. From that gene list, a multivariate predictor is constructed and applied to

the sample that was omitted. The program records whether that prediction was correct or not. This is repeated, omitting all of the samples one at a time. The output table shows which samples were correctly and incorrectly predicted, and the overall cross-validated misclassification rate. The output table is arranged so that you can see the extent to which the different predictors agree or disagree for each specimen left out of the training set. Because of the large number of candidate predictor variables, it is essential to use cross-validation or some similar method to determine whether the model predicts accurately. Even with classes that do not differ in expression profiles, it is very easy to develop models that predict perfectly when measured in a non cross-validated manner. Such models would be useless for application with independent data. For further discussion see Simon R, Radmacher MD, Dobbin K, and McShane LM, Pitfalls in the analysis of DNA microarray data: Class prediction methods, Journal of the National Cancer Institute 95:14-18, 2003.

The options page of the class prediction tool provides alternatives to LOOCV for cross-validation. With k-fold validation, the samples are randomly partitioned into k (approximately) equal size groups. One of the k subsets is omitted and the classifier model developed from scratch using a training set consisting of samples in the union of the other k-1 subsets. The number of errors in using that model to classify samples in the subset omitted is recorded. This is done k times, omitting each of the k subsets one at a time and the total number of errors is obtained. That total number of errors corresponds to that partition into k subsets. The user can indicate that the process should be repeated with different random partitions into k subsets a specified number of times in order to improve the precision of the estimate of cross-validated prediction error. This is called repeated k-fold validation. When the number of samples is small, LOOCV is generally best. When the number of samples is large, LOOCV is very compute intensive and also an alternative like 10-fold cross-validation may provide a more precise estimate. The precision of the estimate does not generally improve greatly with repeating the partition more than once, but the user can experiment with more repeats to ensure that the estimate is stable.

The options page also provides the 0.632+ bootstrap method of re-sampling for estimating prediction error as an alternative to LOOCV or repeated 10-fold cross-validation. It is quite computationally intensive as it involves generating 100 (default) training sets by randomly selecting a subset of the samples to include. It is somewhat unusual in that each sample can be included in a single training set in multiple replicates. The 0.632+ bootstrap method was proposed by B Efron and RJ Tibshirani (Improvements on cross-validation: the .632+ bootstrap method, J. American Statistical Association 92:548-560, 1997) and often performs well. When the number of samples is small, LOOCV is usually satisfactory, however. Cross validation and bootstrap methods for estimating prediction accuracy with high dimensional data have been evaluated and reviewed by AM Molinaro, R Simon and RM Pfeiffer (Prediction error estimation: a comparison of resampling methods, Bioinformatics 21:3301-3307, 2005).

In addition to providing the cross-validated misclassification information, the Class Prediction Tool also (optionally ) provides the permutation p value for the cross-validated

misclassification error rate. That is, for each random permutation of class labels, the entire cross-validation procedure is repeated to determine the cross-validated misclassification rate obtained from developing a multivariate predictor with two random classes. The final p value is the proportion of the random permutations that gave as small a cross-validated misclassification rate as was obtained with the real class labels. There is a cross-validated misclassification rate and a corresponding p value for each class prediction method requested. The user inputs the number of permutations desired. At least 1000 permutations should be specified for a valid permutation p value on the cross-validated misclassification rate. The user may specify only 1 permutation and will quickly obtain the appropriate gene list and valid cross-validated misclassification rates. Only the p values associated with the cross-validated misclassification rates will be missing. Computing the permutation p value for the cross-validated error rate is very compute intensive, because the entire cross-validtion process must be repeated for hundreds or thousands of random permutations of the class labels. Consequently, the default for this option is off, and in many cases it is best to not request this p value until later in the analysis.

The set of "informative genes" , i.e. the genes that are selected as differentially expressed among the classes, will differ for each iteration of the cross-validation. This is because the entire predictor development process must be repeated from scratch for each new cross-validated training set. The gene list provided at the end represents the genes that are selected when the prediction algorithm is applied to the full dataset with no samples omitted. That is the predictor that would presumably be used in the future. But in order to properly estimate the prediction accuracy of that classifier based on the full dataset, one must go thru the cross-validation procedure. The final gene list contains a column labeled "cross-validation support." It indicates the percentage of the cross-validation training sets in which each gene was selected. 100% means that the gene is so strong that it was selected in all of the cross-validated training sets. The initial table in the output, the one showing which samples are correctly and incorrectly predicted by which classifier, has a column indicating how many genes were selected for the cross-validated training set in which each sample was omitted.

Note that the list of genes that are used to form the class predictors will coincide with the list of genes produced by the Class Comparison Between Groups of Arrays Tool for the same analysis between the two classes. The only exception is when there is only one nonmissing value in one of the two classes, or when the variance in one of the two classes is zero. In that case, the compound covariate predictor imputes the p-value for that gene to be 1, rendering that gene nonsignificant for that permutation or analysis, whereas the Class Comparison Between Groups of Arrays Tool does not.

The Class Prediction Tool saves this gene list in the `..\ArrayTools\Genelists\User` folder using the genelist name specified by the user.

## Prediction for new samples

The Class Prediction Tool can be used to classify samples not used in the model building process. In the class prediction dialog, check the box labeled “Use separate test set” . You should create a column in your experiment descriptor worksheet that indicates which experiments contain samples that are to be included in the model building process (labeled “training”), which are to be used only for classification prediction once the model building is completely finished (labeled “predict”), and which samples are to be excluded entirely from both the training and test sets (labeled “exclude”). Predictions will be made for the samples labeled “predict” using all of the class prediction methods requested in the dialog.

Ususally, you do not have a separate set of samples that you wish to withhold from the model building process when the analysis is first done. The model building process itself uses leave-one-out cross validation, and so it is not necessary to have a separate test set. Sometimes, however, after the model is built, you have additional samples whose classes you wish to predict using the previously developed model(s). We have enabled you to do this by re-building the model using the initial set of samples, and to then predict for the new samples in a combined analysis. This has been more direct for us to implement, rather than trying to save the originally developed model, since in cases like k-nearest neighbor classifiers, the entire dataset is needed for classification of new samples.

## Binary tree prediction

The **Binary Tree Prediction Tool** is another algorithm that can be used for class prediction when there are more than 2 classes. The same methods (compound covariate predictor, diagonal linear discriminant analysis, k-nearest neighbor using k=1 or 3, nearest centroid, and support vector machines) form the foundation of both tools. Moreover, for the case of only two distinct classes, binary tree prediction is identical to the Class Prediction Tool (except it evaluates only one prediction algorithm at a time whereas Class Prediction Tool can evaluates all of them in one run). The difference between the algorithms lies in how they treat cases with 3 or more classes.

The Binary Tree Prediction Tool does not attempt to predict the class of a sample in one step. Instead, at each node of a binary tree the samples are classified into two subsets of classes. One or both of the subsets may consist of multiple classes. The user-specified prediction method is applied to construct a predictor that can distinguish the two subsets of classes. The decision on how to split the classes in each node into two subset classes is determined by the split that can be accomplished with the fewest cross-validated misclassification errors. All possible divisions of the classes into two groups are tested, and the best one (with the lowest misclassification rate) is accepted as a node of the binary tree. If the misclassification rate associated with the node is above the specified threshold, the split is not accepted. In this case, the classifier will not attempt to distinguish between the specific classes in the group.

Then, each of the resulting two groups is investigated in a similar manner. The process stops when each group contains only one sample or none of the divisions of the group into the subgroups has the cross-validation misclassification error rate below the threshold.

If cross-validation of the binary tree-prediction algorithm is requested by the user, then the entire process of binary tree building is repeated for each training set, and the overall cross-validation misclassification rate is thereby determined.

Two methods of cross-validation are implemented. For leave-one-out cross-validation method, samples are excluded from the analysis one by one, the remaining samples used to create a classifier that is used to predict a class of the one sample that was set aside. For large data sets, the leave-one-out validation may take very a long time. The tool has an option of using K-fold validation instead of the leave-one-out validation. For K-fold cross-validation method, samples are divided into K approximately equal groups. Then, groups are excluded from the analysis one by one, the samples of the remaining K-1 groups are used to create a classifier that is used to predict a class of the samples that were set aside. The user can assign K. By default, this tool uses leave-one out validation for the data sets with 25 or fewer arrays, and 10-fold validation if the number of arrays exceeds 25.

Some of the algorithms in the standard Class Prediction Tool can be used even when there are more than 2 classes. The sets of genes used in the predictor at each node of the binary tree may differ, however, thus enhancing the ability of the algorithm to discriminate between the classes. More research is needed to investigate relative strength and weaknesses of the binary tree prediction algorithm when compared to the one-step prediction algorithms as applied to the microarray data.

## Prediction analysis for microarrays (PAM)

The **Prediction Analysis for Microarrays (PAM) Tool** represents another method of class prediction, in addition to those provided in the class prediction tool. The method uses the shrunken centroid algorithm developed by Tibshirani et al. (PNAS 99:6567-6572, 2002). The method is similar to the nearest centroid classifier provided in the class prediction tool, but the centroids of each group are shrunken toward each other by shrinking the class means of each gene toward an overall mean. The amount of shrinking is determined by a “tuning parameter” called delta. As the shrinking occurs, some genes will have the same value of shrunken class mean for the different classes, and hence they will have no effect in distinguishing the classes. For larger values of delta, fewer genes will have different shrunken means among the classes, and so the classifier will be based on fewer genes. With this approach, the number of genes included in the classifier is determined by the value of delta. The algorithm provides a k-fold cross-validated estimate of prediction error for all values of delta where k is the minimum class size. The tool indicates the delta corresponding to the smallest cross-validated prediction error and gives the list of genes that are included in the classifier for that value of delta. The

minimum cross-validated prediction error over the range of delta values is however a somewhat biased estimate of the error to be expected with new data. That is because the selection of the optimal delta should be viewed as part of the classification algorithm and should be included in the cross-validation procedure. We have utilized the PAM analysis program developed by Tibshirani, Hasti, Narashimha and Chu, however, and it does not incorporate the optimization of delta within each step of the cross-validation. The bias may not be large in cases where the graph of cross-validated error as a function of delta is relatively flat, and that graph is provided as part of the output.

## Survival analysis

The Survival Analysis Tools are for analyzing time to event data, where some of the observations are right censored; that is, the survival time for some of the individuals is known to be at least as large as the data recorded, and the patient was still alive at the time of last follow-up. The survival analysis tools can be used for any such time to event data, but we shall refer to such data as "survival" data. Two tools are provided. One tool finds genes that are correlated with survival time for patients. There are many statistical methods for analysis of censored survival data. The most popular method is Cox's proportional hazards model, and the Efron method of handling ties is implemented here. This is a regression model in which the hazard function for an individual is a function of predictor varE iables. In our case the predictor variables are log expression levels. The hazard function is the instantaneous force of mortality at any time conditional on having survived till that time. The proportional hazards model postulates that the logarithm of the hazard of death is a linear function of the predictor variables, linked by unknown regression coefficients. For more details see biostatistics texts or the original paper (DR Cox, Regression models and life tables, J.Royal Stat Soc B 34:187-202).

The Survival Analysis Tool for finding genes whose expression is correlated with survival time fits proportional hazards models relating survival to each gene, one gene at a time and computes the p value for each gene for testing the hypothesis that survival time is independent of the expression level for that gene. Gene lists are created based on these p values in the same way as in the Class Comparison tools. The p values can be used to construct gene lists using multivariate permutation tests for controlling the number or proportion of false discoveries. Or the gene list can simply consist of the genes with p values less than a specified threshold (0.001 is default). For more information regarding the multivariate permutation tests for controlling the number or proportion of false discoveries, please see the preceding section on Multivariate Permutation Tests for Controlling Number and Proportion of False Discoveries.

The Survival Analysis Prediction Tool develops a gene expression based predictor of survival risk group. The number of risk groups and the risk percentiles for defining the groups are specified by the user. The survival risk groups are constructed using the supervised principal component method of E Bair and R Tibshirani (2004):

E Bair & R Tibshirani, Semi-supervised methods to predict patient survival from gene expression data, PLOS Biology 2:511-522, 2004).

This method uses a Cox proportional hazards model to relate survival time to k "super-gene" expression levels, where k is selectable by the user (usually 1-3). The "supergene" expression levels are the first k principal component linear combinations of expression levels of the subset of genes that are univariately correlated with survival. The user specifies the threshold significance level (e.g. 0.001) for selecting the genes to be used in computing the principal components. The significance of each gene is measured based on a univariate Cox proportional hazards regression of survival time versus the log expression level for the gene. After selecting the genes, the principal components are computed, and the k-variable Cox proportional hazard regression analysis is performed. This provides a regression coefficient (weight) for each principal component.

Having developed a supervised principal component model as described above, to compute a prognostic index for a patient whose expression profile is described by a vector x of log expression levels, the following steps are performed. First the components of the vector x corresponding to the genes that were selected for use in computing the principal components are identified. Then the k principal components are computed. These are just linear combinations of the components of x, with the weights of each linear combination having been determined from the principal component analysis described above. Finally, the weighted average of these k principal component values is computed, using as weights the regression coefficients derived from the k-variable Cox regression described above. That computation provides a prognostic index for a patient with a log expression profile given by a vector x. A high value of the prognostic index corresponds to a high value of hazard of death, and consequently a relatively poor predicted survival.

In order to evaluate the predictive value of the method, Leave-One-Out-Cross-Validation is used. A single case is omitted and the entire procedure described above is performed to create a prognostic index. This function is created from scratch on the training set with the one case removed, including determining the genes to be included in the calculation of the principal components. Having determined a prognostic index function for that training set, it is used to compute a prognostic index for the omitted observation. That value is compared to the prognostic index for the n-1 cases included in that training set (assuming that there are n distinct cases available in total). The prognostic index for the omitted patient is ranked relative to the prognostic index for the patients included in the cross-validated training set. The omitted patient is placed into a risk group based on his/her percentile ranking, the number of risk groups specified, and the cut-off percentiles specified for defining the risk groups. This analysis is repeated from scratch n times, leaving out a different case each time.

Having completed the computations described in the previous paragraph, we plot Kaplan-Meier survival curves for the cases predicted to have above average risk and the cases predicted to have below average risk. It is important to note that the risk group for each case was determined based on a predictor that did not use that case in any way in its construction. Hence, the Kaplan-Meier curves are essentially unbiased and the separation

between the curves gives a fair representation of the value of the expression profiles for predicting survival risk.

The Survival Risk Group Prediction tool also provides an assessment of whether the association of expression data to survival data is statistically significant. A log-rank statistic is computed for the cross-validated Kaplan-Meier curves described above. Let $LR_d$ denote the value of that log-rank statistic computed for the data. Unfortunately, this log-rank statistic does not have the usual chi-squared distribution under the null hypothesis. This is because the data was used to create the risk groups. We can, however, obtain a valid statistical significance test by randomly shuffling the survival data among the cases and repeating the entire cross-validation process. For each random re-shuffling, we repeat the process, create new cross-validated Kaplan-Meier curves, and compute the log-rank statistic for the random shuffling. This provides a null-distribution of the log-rank statistic created in this way. The tail area of this null distribution beyond the value $LR_d$ obtained for the real data is the permutation significance level for testing the null hypothesis that there is no relation between the expression data and survival.

The Survival Risk Group Prediction tool also lets the user evaluate whether the expression data provides more accurate predictions than that provided by standard clinical or pathological covariates or a staging system. The user specifies the columns of the experimental descriptor worksheet that defines the clinical covariate. This could be a single column specifying a stage or a set of columns for which a multivariate proportional hazards model will be developed. Kaplan-Meier curves are developed for the covariates without any use of the expression data. An additional model is also developed for a combination of the covariates and the expression data. For each cross-validated training set, genes are selected which add to predicting survival over the predictive value provided by the covariates for that training set. The principal components of those genes are computed and a model fitted containing the covariates and the supervised principal components. The survival risk group for the patient(s) omitted from that training set are predicted using that composite model. In that way cross-validated Kaplan-Meier curves are developed for the combination of covariates and expression data. A permutation analysis is performed in which the survival data and covariates are kept together for each patient but the expression profile is permuted. The cross-validated Kaplan-Meier cuves and log-rank statistics are generated for those permutations and finally a p value is determined which measures whether the expression data adds significantly to risk prediction compared to the covariates.  This approach to combining covariates and expression data is experimental and may be modified or supplemented in future releases.

Version 3.6, has an additional tool called Survival Gene Set analysis. Similar to the gene set comparison tool, this tools finds sets of genes that are correlated with survival. A proportional hazards model is fitted to survival time, one gene at a time and the corresponding p value for the gene is computed. The LS and KS statistics are computed using the p-values from the proportional hazard model for each in a gene set. The tests are applied separately to each gene set in the category. A gene set is selected to be significant if its LS or KS re-sampling p-value is below the threshold specified by the user. Results are provided as a table of selected gene sets that are ordered by the p-value

of the LS test. For each gene set, the table lists the unique identifier, the number of genes in the project gene list that belong to that gene set, the LS p-value and the KS p-value. Another table lists all the genes that are found in the selected gene sets.

For BRB-ArrayToolsv3.7 a new gene set analysis method, Efron-Tibshirani's Gene Set Analysis (GSA, Efron & Tibshirani 2007), is implemented to use "maxmean" statistics improved upon the original Gene Set Enrichment Analysis (GSEA) procedure of Subramanian et al. (2005) for assessing signficance of pre-defined gene-sets. GSA uses the maxmean statistic: this is the man of the positive or negative part of gene scores di in the gene set, whichever is large in absolute value. Efron and Tibshirani shows that this is often more powerful than the modified Kolmogorov-Smirnov statistic used in GSEA. The R-package GSA is automatically imported by ArrayTools and used for calculation.

The method described above can be used to analyze GO Categories, BioCarta pathways, KEGG pathways, microRNA targets, transcription factor targets, protein domains, the Broad Institute's MSigDB gene sets collections as well as user-defined genelists.

# Quantitative traits analysis

This tool finds genes that are significantly correlated with a specified quantitative variable (trait) such as age. Spearman or Pearson correlation coefficients are used as a measure of correlation and to compute parametric p-values. Most of the options are identical to the Class Comparison Tools. Two exceptions are the pairing of the samples and the random variance model that are not implemented for the Quantitative Traits Analysis Tool. Output of the tool is also nearly identical to that for the Class Comparison Tools.

# Some options available in classification, survival, and quantitative traits tools

## Random Variance Model

The random variance model is found as an option in the class comparison and class prediction tools. The *random variance* t and F tests can be selected as alternatives to the standard t/F tests. The standard t/F tests are based on the assumption that the within class variance is different for different genes and hence these variances are estimated separately for each gene. When there are few samples in each class, the variance estimates are not very accurate and the statistical power of t/F tests are poor. Some published methods assume that the variances are the same for all genes, and that is clearly a poor assumption. The random variance model takes an intermediate approach. It assumes that different genes have different variances, but that these variances can be regarded statistically as independent samples from the same distribution. Specifically, the reciprocal of the variances are regarded as draws from a gamma distribution with two parameters (a,b). The parameters are estimated from the complete set of expression data

and we first test that the model is accurate. In most datasets we have examined, the model assumption is very accurate. The usual t test for comparing expression of gene i in two classes is based on the statistic

$$t = \frac{\overline{x}_{i,1} - \overline{x}_{i,2}}{\hat{\sigma}_i \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

where the numerator is the difference in the means of log expression for gene i in the two classes, $n_1$ and $n_2$ are the number of samples in the two classes, and $\hat{\sigma}_i$ denotes the square root of the usual within-class variance for gene i. The within-class variance is computed by a weighted average of the variance of log expression in each of the two classes. With the usual t test, under the null hypothesis, the t value has a t distribution with $n_1+n_2-2$ degrees of freedom. For the random variance model, the same formula for t is used, except that $\hat{\sigma}_i$ is replaced in the denominator by $\tilde{\sigma}_i$ where

$$\tilde{\sigma}_i^2 = \frac{(n_1 + n_2 - 2)\hat{\sigma}_i^2 + 2a\left(1/ab\right)}{(n_1 + n_2 - 2) + 2a}$$

The quantity 1/ab is an estimate of the expected variance for the inverse gamma model. Consequently, $\tilde{\sigma}_i^2$ is a weighted average of the usual gene specific variance and the average variance for all the genes. The weight given to the gene specific variance is the total number of samples in the two classes minus two. The weight given to the average variance for all the genes is 2a. It can be shown rigorously that when the model assumption holds, that the t values computed using $\tilde{\sigma}_i$ have a t distribution under the null hypothesis but the degrees of freedom is now n-2+2a. In cases where the number of samples n is small, this increase in degrees of freedom and the associated improvement in the estimation of variances results in improved statistical power for detecting differentially expressed genes. These results extend directly to the F distribution for comparing more than two classes. For details, see G Wright and R Simon, The random variance model for finding differentially expressed genes, Bioinformatics (19:2448-2455, 2003), also Technical Report, Biometric Research Branch, National Cancer Institute, http://linus.nci.nih.gov/~brb ; P Baldi and AD Long, A Bayesian framework for the analysis of microarray expression data: regularized t-test and statistical inferences of gene changes, Bioinformatics 17:509-519, 2001.

## Multivariate Permutation Tests for Controlling Number and Proportion of False Discoveries

The multivariate permutation tests for controlling number and proportion of false discoveries is an option found in the class comparison, survival analysis, and quantitative traits analysis tools. Using a stringent p<0.001 threshold for identifying differentially expressed genes is a valid way for controlling the number of *false discoveries*. A false

discovery is a gene that is declared differentially expressed among the classes, when in fact it is not. There are two problems with this approach to controlling the number of false discoveries. One is that it is based on p values computed from the parametric t/F tests or random variance t/F tests. These parametric p values may not be accurate in the extreme tails of the normal distribution for small numbers of samples. The second problem is that this approach does not take into account the correlation among the genes. Using stringent p value thresholds on the univariate permutation p values won't be effective when there are few samples and will not account for correlations. We have implemented multivariate permutation tests that accomplish both objectives. They are described in detail in R Simon, EL Korn, LM McShane, MD Radmacher, GW Wright and Y Zhao, *Design and Analysis of DNA Microarray Investigations*, Springer 2003, in EL Korn, JF Troendle, LM McShane and R Simon, Controlling the number of false discoveries: Application to high dimensional genomic data; Journal of Statistical Planning and Inference (In Press), also Technical Report 3, Biometric Research Branch, National Cancer Institute, 2002; http://linus.nci.nih.gov/~brb. This approach is also described by Reiner A, Yekutieli D and Benjamini Y, Identifying differentially expressed genes using false discovery rate controlling procedures, Bioinformatics 19:368-375, 2003).

The multivariate permutation tests are based on permutations of the labels of which experiments are in which classes. If there are fewer than 1000 possible permutations, then all permutations are considered. Otherwise, a large number of random permutations are considered. For each permutation, the parametric tests are re-computed to determine a p value for each gene that is a measure of the extent it appears differentially expressed between the random classes determined by the random permutation. The genes are ordered by their p values computed for the permutation (genes with smallest p values at the top of the list). For each potential p value threshold, the program records the number of genes in the list. This process is repeated for a large number of permutations. Consequently, for any p value threshold, we can compute the distribution of the number of genes that would have p values smaller than that threshold for permutations. That is the distribution of the number of false discoveries, since genes that are significant for random permutations are false discoveries. The algorithm selects a threshold p value so that the number of false discoveries is no greater than that specified by the user C% of the time, where C denotes the desired confidence level. If C=50%, then the median number of false discoveries is that specified by the user. If, for example, C=90%, then you have 90% confidence that the number of false discoveries is no greater than that number specified by you. In a similar manner, we determine threshold p values so that the resulting gene list contains no more than a specified proportion of false discoveries.

The class prediction tool produces a gene list ordered with the genes having the smallest parametric p values at the top. The length of the gene list is determined by the types of false discovery control selected. Generally, we recommend using all of the options: univariate p value threshold (0.001); limiting number of false discoveries (10 default), and limiting proportion of false discoveries (0.10 default). The output tells you where to truncate the gene list in order to get each type of control.

The procedures for controlling the number or proportion of false discoveries are based on multivariate permutation tests. Although parametric p values are used in the procedures, s the permutation distribution of these p values is determined, and hence the false discovery control is non-parametric and does not depend on normal distribution assumptions.

The multivariate permutation tests use the expression data much more efficiently than the univariate permutation tests. Consequently, we no longer recommend using the univariate permutation tests. We have retained the option to do the univariate permutation tests, but we have made the default to skip them. The multivariate permutation tests are particularly more effective when the number of samples is limited. In that case it may be impossible to obtain differences significant at the 0.001 level by a univariate permutation test, but quite possible to find significant genes while ensure that the number or proportion of false discoveries is controlled using the multivariate permutation tests.

The multivariate permutation tests take advantage of the correlation among the genes. For a given p value for truncating an ordered gene list; the *expected number* of false discoveries does not depend on the correlations among the genes, but distribution of the number of false discoveries does. The distribution of number of false discoveries is skewed for highly correlated data. If you specify the confidence coefficient at 50%, the program provides the length of the gene list associated with a specified median number of false discoveries or given proportion of false discoveries. You may think of those gene lists as being expected to contain the target number or proportion of false discoveries, although the median and expected number are not exactly the same. Requiring 90% confidence that the gene list does not contain more false discoveries than the target amount is a more stringent condition and the associated gene lists will be much shorter than those based on the median. When there are few samples, the median number of false discoveries may be highly variable, and we recommend using the 90% confidence level.

The HTML output file created by the Class Comparison Tools gives the parametric p-value, the univariate permutation p-value (if requested), and the geometric mean ratio (geometric mean intensity for single color oligonucleotide arrays) in each class, and gene identifying information. The list is sorted by parametric significance level. If there are two classes, the list is in two parts; those genes down-regulated in class one relative to class two followed by those genes up-regulated in class one relative to class two. When there are more than two classes, the patterns of differential expression among the classes will be more varied for genes in the list. The Class Comparison Tools save this gene list in the `..\ArrayTools\Genelists\User` folder using the genelist name specified by the user. It is often useful to cluster the samples with regard to this gene list in order to better understand the pattern of genes across samples.

## Specifying replicate experiments and paired samples

For the class comparison and class prediction analysis tools, the user may need to specify replicate experiments or paired samples, and for the survival analysis and quantitative traits analysis tools, the user may need to specify replicate experiments on the same

sample. To enter paired samples into an analysis, the user must create a descriptor variable on the Experiment descriptors sheet in which a unique value is given for each pair. The following example shows the experiment design sheet for a paired analysis:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Array_Id | Patient_Id | Treatment | | |
| 2 | Array001 | Patient001 | Pre-Treat | | |
| 3 | Array002 | Patient001 | Post-Treat | | |
| 4 | Array003 | Patient002 | Pre-Treat | | |
| 5 | Array004 | Patient002 | Post-Treat | | |
| 6 | Array005 | Patient003 | Pre-Treat | | |
| 7 | Array006 | Patient003 | Post-Treat | | |
| 8 | Array007 | Patient004 | Pre-Treat | | |
| 9 | Array008 | Patient004 | Post-Treat | | |
| 10 | Array009 | Patient005 | Pre-Treat | | |
| 11 | Array010 | Patient005 | Post-Treat | | |
| 12 | Array011 | Patient006 | Pre-Treat | | |
| 13 | Array012 | Patient006 | Post-Treat | | |
| 14 | Array013 | Patient006 | Post-Treat | | |
| 15 | | | | | |
| 16 | | | | | |

Experiment descriptors / Gene identifiers / Filtered log ratio

Here, the pre-treatment samples are to be compared against the post-treatment samples for each patient. The pairing variable is `Patient_Id` and the class variable to be compared is `Treatment`. Note that both `Array012` and `Array013` contain post-treatment samples for `Patient006`. In that case, the average of the two experiments will be used for the analysis.

When an unpaired analysis is used, the user should make sure to specify whenever there are replicate experiments that have been performed using the same sample. The following example shows an experiment descriptor sheet in which replicate experiments were performed for some of the samples:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Array_Id | Sample_Id | Phenotype | | |
| 2 | Array001 | Sample001 | Primary | | |
| 3 | Array002 | Sample001 | Primary | | |
| 4 | Array003 | Sample002 | Metastasis | | |
| 5 | Array004 | Sample002 | Metastasis | | |
| 6 | Array005 | Sample003 | Primary | | |
| 7 | Array006 | Sample004 | Primary | | |
| 8 | Array007 | Sample005 | Metastasis | | |
| 9 | Array008 | Sample006 | Primary | | |
| 10 | Array009 | Sample007 | Metastasis | | |
| 11 | Array010 | Sample008 | Primary | | |
| 12 | Array011 | Sample009 | Primary | | |
| 13 | Array012 | Sample010 | Metastasis | | |
| 14 | Array013 | Sample011 | Metastasis | | |
| 15 | | | | | |
| 16 | | | | | |

Experiment descriptors / Gene identifiers / Filtered log ratio

In this case, `Sample_Id` is the replicate variable to average over. Since `Array001` and `Array002` have the same `Sample_Id`, their average will be used for the analysis. Similarly, `Array003` and `Array004` will be averaged for the analysis. All the other experiments will be used as-is, since they have no replicates.

## Gene Ontology observed v. expected analysis

The Gene Ontology Observed vs. Expected analysis is an option found in the class comparison, class prediction, survival analysis, and quantitative traits analysis tools.

The purpose of the Gene Ontology Observed vs. Expected analysis is to give information whether or not the list of significant genes selected by the analysis is different from a genelist randomly selected from all genes in the analysis, for a given Gene Ontology. This is expressed by the Observed vs. Expected ratio. Before we define the Observed and Expected ratio, the term “genes in the analysis” should be clarified. The “genes in the analysis” refers to the set of genes which were input into the class comparison, class prediction, survival analysis, and quantitative traits analysis tool, i.e. the genes passing the filtering criteria.

The **Observed** is defined as the number of genes in the list of significant genes selected by the analysis tool which fall into a Gene Ontology category. The **Expected** is defined as the average number of genes which would be expected to fall into that Gene Ontology category in a subset of genes randomly selected from the genes in the analysis. A Gene Ontology category is considered to consist of not only the genes which are described by

that Gene Ontology term, but also any gene which is described by any children of that Gene Ontology term.

For example, suppose we are computing the observed and expected counts of the Cellular Component term "Cytoplasm". We choose a set of genes, denoted as A, which has total N number of genes with a Cellular Component term, in the analysis of Class Comparison. From these N genes, we get n significant genes by certain criteria, which forms a subset S. In the set A, there are M genes belonging to the Cellular Component term "Cytoplasm" or any of the term's children. And in the subset S, there are m genes belonging to the Cellular Component term "Cytoplasm" or any of the term's children. Then, the Observed is m and the Expected is $n \times M/N$. The ratio of Observed vs. Expected for "Cytoplasm" is $m/(n \times M/N)$, simplified by $mN/Mn$.

The user may specify a threshold for the minimum number of genes which must be in a Gene Ontology category to be considered (default=5), as well as a threshold for the minimum Observed vs. Expected ratio to be considered (default=2) in the **Options** page of the analysis dialogs.

# Programmable Plug-In Faciltiy

BRB-ArrayTools has a plug-in facility that enables users or methodology developers to extend the tool set built in to BRB-ArrayTools by providing their own tools. Users can then easily add-in the new tools to their version of BRB-ArrayTools and use them in the analysis of their data. The extensions must be written as functions in the R statistical programming language. R is an open source and very powerful statistical language that is popular with academic statistical scientists. It is in many ways a public domain version of S+, although each language has some features missing in the other. There is a growing body of R functions developed for the analysis of microarray data and the plug-in facility can give BRB-ArrayTools users and R software developers access to each other via BRB-ArrayTools.

Detailed information about the creation and use of add-in R functions is described in a separate document entitled BRB-ArrayTools Plug In Guide. Here we will give an overview of the plug-in facility. In considering R plug-ins there are two perspectives, one for the R tool developer and one for the BRB-ArrayTools user analyzing his/her data and wanting to utilize plug-ins.

For the plug-in user, the process of installing and accessing a plug-in is quite simple. The BRB-ArrayTools main menu contains a "Plug In" entry and a "Load Plug In" sub-menu entry. The Load Plug In dialog box askes for the path name of the plug in and it asks whether the user wants the plug-in added to the plug-in menu for future use. When the user clicks the "Load" button on the Load Plug In dialog box, the plug-in is launched and it will prompt the user for any further information it needs. BRB-ArrayTools will pass the R plug-in any data that it needs in the user's Project Workbook that has already been collated by the user. The output of the analysis performed by the plug-in will be directed

to a file and that file will be opened either by the plug-in itself or by the user. The user only needs to know the name of the plug-in. In using plug ins obtained from others, the user has to install the plug in in the PlugIns folder of the BRB-ArrayTools directory. There is a .r sub-folder for r functions, a .plug sub-folder for plug-in interfaces created by developers, and a .txt sub-folder for readme files created by developers.

The creation of R plug-ins for BRB-ArrayTools is also relatively easy. The first step is to write the R function that performs the desired analysis. This function will be launched by the BRB-ArrayTools user as described above, and BRB-ArrayTools will pass to the R function the information in the active collated workbook that the function needs. The data will be automatically and the developer need not read any files to obtain the desired data. In writting the R function, the developer can name the available data objects provided by BRB-ArrayTools any way he/she chooses. Because the R function runs in the background, output from the function should be written to a file. The R function may open the file before terminating or the user can open the file.

The second step in creating a plug-in is to create the interface between the R function and BRB-ArrayTools. We provide a wizard to make this step easy. The wizard enables the developer to tell BRB-ArrayTools what data objects the R function needs and what names the R function wants associated with those data objects. The wizard also enables the developer to prompt the user for additional data input. The input dialog will be in the format of the Micrososft Windows type of dialog boxes used by BRB-ArrayTools but the VBA coding of these dialogs will be automatically generated by BRB-ArrayTools using information provided by the developer in his/her dialog with the wizard. For more information, see the BRB-ArrayTools Plug In Guide.

# Pre-installed plugins

## Analysis of variance

BRB-ArrayTools contains several ANOVA plugins:

- Basic ANOVA
- ANOVA for mixed effects model
- ANOVA on log-intensities
- Time series regression analysis

## Random forest

Random forest is a class prediction method which bases predictions on majority voting of a collection of decision trees.

## Top scoring pair class prediction

The top scoring pair class prediction is a method for finding the pair of genes whose expression best classifies samples into 2 pre-defined classes.

## Sample Size Plug-in

The Sample Size plug-in calculator provides information on the number of samples per class needed for identifying genes that are differentially expressed between the classes when controlling for the number of false positives.

## Nonnegative matrix factorization for unsupervised sample clustering

This method has been developed as an unsupervised sample clustering of microarray gene expression data. *Nonnegative matrix factorization for unsupervised sample clustering*

# Further help

## Some useful tips

### Utilities

#### Preference Parameters

To modify some of the default preference parameters, from the “ArrayTools” pull down menu, select the option “Utillities” and the choose the “Preference”. The allows you to turn and off the following parameters:

Variable Value Comments

LowessSpan 0.4 #Span for lowess parameter (proportion of data used for each point in lowess curve).

ShowWelcome True #Shows the Welcome screen when BRB-ArrayTools is loaded.

DebuggerOn True #Turns on debugger which points to offending line of code.

PluginsUpgrade 3.1 #Plugin files have been upgraded to the structure of the version listed in this parameter.

CheckedUpdate 3/19/2009 #Date of last update check.

UpdateInterval 7 #Number of days to wait before checking for updates again. Use FALSE to turn off updates feature.

DendroWidthG 580 #Pixel-width of dendrogram for clustering genes (in pixels). (Or enter "default" to use default value of 580.)

DendroHeightG 400 #Height of dendrogram for clustering genes (in pixels). (Or enter "default" to use default value of 400.)

DendroWidthE Default #Pixel-width of dendrogram for clustering experiments (in pixels). (Or enter "default" to use default values customized to plot.)

DendroHeightE Default #Height of dendrogram for clustering experiments (in pixels). (Or enter "default" to use default values customized to plot.)

ChangeRServer False #If TRUE, then use RExcel.xla instead of RServer.xla.

PercentMissing 50 #If total % missing > PercentMissing, then pass data as character not numeric, to avoid time spent in converting missing values.

Verification True #If TRUE, then turn on the data import, filtering, and normalization error detection.

RetainTempFiles False #If TRUE, then turn on to retain the temporary files in the working path folder.

StopCheckAnnotationsFolder False #If TRUE, then stop checking if project contains annotations folder.

Heatmapthreshold 3 #Truncate parameter for log-ratio or intensity of heatmap. These parameters apply to analysis related heatmaps for Class Comparison, Geneset Class Comparison, and Time Course Analysis tools. For the dual channel data, the “Threshold log-ratio = X” is the thresholding parameter for the log-ratio values. Any values larger than X will be assigned to be X, and any values less than –X will be assigned to be –X. Minvalue is the color for the minimum value, MidValue is the color for the middle value (usually 0), and MaxValue is the color for the maximum value.

For the Single channel data, the “Threshold log-intensity percentile”, a lower percentile from X1 to a higher percentile X2 are used for thresholding the data. The program will calculate the intensity values I1 and I2 of the percentiles X1 and X2. Any value larger than I2 will be assigned to be I2 and any values less than I1 will be assigned to be I1. MinVaule is the color for the minimum value, and MaxValue is the color for the maximum value.

## Download packages from CRAN and BioConductor

If you plan to run various analysis tools on your collated project data offline (without an internet connection), we have added a utility feature to pre-download all the packages necessary to run various analysis tools and plug-ins from the CRAN site as well as Bioconductor. You must be connected to the internet to run this utility. The following is a list of packages that get downloaded.
Packages downloaded from BioConductor are:
affy
annotate
annaffy
gcrma
globaltest
GO.db
lumi
ROC
simpleaffy

Packages downloaded from CRAN:
bitops
car
gplots
GSA
impute
lars
matlab
pamr
randomForest
R2HTML
RColorBrewer
XML
xtable

## Excluding experiments from an analysis

Some of the analyses within BRB-ArrayTools can be run on a subset of the experiments in the collated project workbook. For the hierarchical clustering and multidimensional scaling tools, an experiment will be excluded from the analysis if it has a blank label

within the designated experiment descriptor variable. For the scatterplot of phenotype averages and the classification tools, an experiment will be excluded from the analysis if it has a blank label within the selected phenotype class variable.

## Extracting genelists from HTML output

Sometimes the user may need to extract columns of data from the HTML tables that are produced by BRB-ArrayTools. For example, the user might wish to include the HTML genelist table produced by the classification tools in a Word document for publication. Or the user might need to extract the t-values and midpoint values from the compound covariate predictor output for use in classifying new samples. Or the user might wish to sort the genelist table by a particular column.

HTML tables can be easily converted to tables in Word or columns in Excel. You can do this in one of two ways. One way is to select the HTML table by left-clicking and dragging the mouse over the entire table, and then copying to the buffer through **Edit → Copy** (or **Ctrl-C**). Then open up a blank Word or Excel document, and paste the contents of the buffer into the document through **Edit → Paste** (or **Ctrl-V**).

Another way is to open the HTML file within Word or Excel using the **File → Open** menu. In the **Open** dialog box, go to the **Files of type** field and select **HTML Document** (in Word or Excel 97) or select either **All Microsoft Excel Files** or **Web Pages** (in Excel 2000). Now browse for the HTML document which you would like to convert. Once the table has been converted to a Word table or to cells within Excel, it is very easy to add to, delete, or edit any of the rows or columns in the table. Note that all hyperlinks are preserved when converting HTML tables to Word or Excel.

## Creating user-defined genelists

The best utility to use for creating user-defined genelists is Notepad. This is because Notepad always saves files as ASCII text files, without adding the hidden formats into the file which some word processors do. To open Notepad, click on **Windows Start Button → Programs → Accessories → Notepad**.

To save any list of genes from a column of cells within Excel, simply select the cells by left-clicking and dragging your cursor over the cells, and copying to the buffer through **Edit → Copy** (or **Ctrl-C**). Then open up a blank text file in Notepad, and paste the contents of the buffer into the text file through **Edit → Paste** (or **Ctrl-V**). It is possible to save the list of selected genes from a BRB-ArrayTools scatterplot in this way.

To save any list of genes from an HTML table, first convert to Word or Excel, then copy the desired column of gene identifiers into a Notepad document. Of course, the genelists from the Class Comparison or the Compound Covariate Predictor tool are already automatically saved into the `..\ArrayTools\Genelists\User` folder.

However, the user may be interested in saving the genes from an HTML table for purposes other than to enable subset analyses within BRB-ArrayTools, since the HTML tables report the genes in sorted order by p-value or another criterion.

A new option in BRB-ArrayToolsv3.4.0 called "Create Genelist using GO" allows the user to enter a search string for Gene Ontology terms. The output from the search, is a gene list file which will contain the list of unique Ids where the search criteria term matches the GO description. The user can name the gene list file appropriately and this list gets saved in the "Genelists" folder under the user's project folder. The genelist so created will now be available for the user to use by running the GeneSubset option from the Filter and Subset dialogue.

Another utility feature added in BRB-ArrayTools 3.6, is called "create genelist"->"Correlated with a target gene". This utility is for finding genes positively and negatively correlated with a target gene above a minimum threshold value. The inputs are the type of gene identifier, the value of gene identifier for a target gene, the minimum correlation threshold (default 0.5), and the name for the output genelist folder. This utility uses only the genes that have passed the filtering criteria. The output is a genelist file containing multi-column identifiers for those genes that are correlated above the specified correlation threshold with the target gene. This output generates a HTML table showing the positive and negatively correlated genes and generates corresponding genelist files that are placed in the project folder's "genelists" subfolder. The resulting genelist will be used with in the "Gene subsets" of the "Filter and subset the data" option. If duplicates of a target gene are found, the averaged value for the target gene will be used in computing correlation with every gene else.

New to v4.1.0, is the option to obtain drug names based on any genelist. The output from this a list sorted by either gene names or drug names. The link that the data is obtained from is http://www.drugbank.ca/. The DrugBank database is a unique bioinformatics and cheminformatics resource that combines detailed drug (i.e. chemical, pharmacological and pharmaceutical) data with comprehensive drug target (i.e. sequence, structure, and pathway) information. The database contains nearly 4800 drug entries including >1,350 FDA-approved small molecule drugs, 123 FDA-approved biotech (protein/peptide) drugs, 71 nutraceuticals and >3,243 experimental drugs. Each DrugCard entry contains more than 100 data fields with half of the information being devoted to drug/chemical data and the other half devoted to drug target or protein data.

## Affymetrix Quality Control for CEL files:

This utility which generates quality control plots and RNA degradation plots has been included in v3.5.To run the utility, you must open a project that has been collated using .CEL files and select the column from your experiment descriptor file that you wish to use as the column defining your arrays. This utility uses functions from the Bioconductor libraries 'simpleaffy' and 'affy'. The HTML output contains plots and associated tables. The main objective is to identify any outlier arrays within a class. The output contains Quality Control(QC) plots for each class variable for the following metrics: Scale factor,

average background noise, percentage of genes called present, 3'to 5' ratios for GAPDH and B-actin ratios.
The typical recommendations are as follows:
**GAPDH 3':5' ratios**: The ratio values should be around 1, values that are larger than 1.25 are colored red.
**B-actin 3':5` ratios:** These ratios should be 3, values that are smaller than 3 are colored in blue and values that are above 3 are colored red as it may indicate an issue with labeled RNA.
**Scale Factor**: The blue strip in the image represents the range where the scale factors are within 3-fold of the mean for all the chips. If any scale factors fall outside this range they are colored red.
**Average Background:** This value is represented by a number besides the image and these values should be consistent with each other. Large variations in these values are colored red, otherwise in blue.
**Percentage Present:** The percentage of genes called present is shown besides the image and these values too should be consistent with each other.
Additionally, a table of the QC metrics for each array is presented below the plot. In addition to the QC metric plot, a RNA degradation/digestion plot is also provided for all the arrays within the class. The objective of this plot is to assess the quality of the RNA used in each sample. The plot, shows the expression as a function of 5'-3' position of probes. For each array within each probeset , probes are arranged by their proximity to the 5' end of the gene. The plot shows the average intensity of the probes classified by this order. Only the PM probes are taken into account for this plot. Each line represents an array and usually we expect the lines to be parallel. Plots showing lines with different trends could indicate possible differences in laboratory processing. Below the plot, a table containing the slopes and p-values of the slopes is also provided.
For additional details refer to
http://bioconductor.org/packages/1.8/bioc/vignettes/simpleaffy/inst/doc/simpleAffy.pdf

## Using the PowerPoint slide to re-play the three-dimensional rotating scatterplot

When running the three-dimensional rotating scatterplot, users who have PowerPoint 2000 or later installed will also have a PowerPoint file created in the `Output` folder of the project folder. The name of this file is specified on the **Options** page of the **Multidimensional scaling of samples** dialog, and is named `MDS.ppt` by default, unless the user changes this name. To play the three-dimensional rotating scatterplot from the PowerPoint file, open up the PowerPoint file, and choose to **Enable macros** when prompted. Once the slide is opened, go into **Slide Show** mode by clicking on the **View → Slide Show** menu item in PowerPoint. Once you are in **Slide Show** mode, then click on the **Click Here to Display** link to view the rotating scatterplot. You may edit the slide in **Edit** mode (such as changing the title, removing the warning label, or changing the caption on the link by right-clicking on it and opening the **Properties** dialog). However, the **Click Here** link will open up the rotating scatterplot only if you click on it during the **Slide Show** mode and not during the **Edit** mode. To go into the **Edit** mode, click on the **View → Normal** menu item in the PowerPoint menu.

Users who do not have PowerPoint 2000 or later will not have this PowerPoint file created for them. However, these users may still save a screenshot of the scatterplot or re-play the rotating scatterplot from a DOS command, as described below. To save a still-shot of the screen, simply press the **PrintScreen** button on the keyboard, then go into any application which has image editing capabilities, and paste the screenshot (usually by pressing the **Ctrl-V** key combination).

## Stopping a computation after it has started running

Sometimes a user may want to cancel a computation after it has started running. If the computation is running in Excel, then the user may stop the computation from within Excel by pressing the **ESC** key, or by pressing the **Ctrl-Break** key combination. If the computation is running as a batch job within an MS-DOS window, then the computation may be cancelled by simply closing the MS-DOS window.

## Automation error

An automation error is an error which occurs when one of Excel's built-in functions is unable to run. This error often occurs at the save step, when the project workbook is unable to be saved to the disk for some reason. One possible cause for this error is that a user's disk may already be full. However, many users have encountered this error during the collation step, even when their disks had not been full. The exact cause of this error is not yet known, as the error is often not reproducible on another computer system running the same software when collating the same dataset. However, a workaround does exist which enables users to continue using their project workbook even when this error does occur.

The project workbook is saved to disk twice during the collation step. Before the collation process actually begins, a template project is saved to disk, containing all the proper worksheets but no data. After the collation process is completed, the project is re-saved to the disk, overwriting this original template. When the automation error occurs during the second save step, the user can often still open the original template project workbook and use it to regenerate all the data. Because the collation process did finish before the automation error at the second save step, all the data has already been written to binary files in the project folder. After opening the empty project workbook, the user may force the project to regenerate all the data by re-filtering the project using a slightly different set of spot filtering criteria (for example, by changing the intensity threshold). The re-filtering process will regenerate all the data which had already been written to binary files during the collation step, and the user may then change the filters back to their original settings, re-filter again, and then save the project workbook manually.

## Excel is waiting for another OLE application to finish running

Sometimes when a user is running a large computation (usually occurs if the user is running a hierarchical clustering of more than approximately 4000 genes), the user may receive the following message from Excel: "Excel is waiting for another OLE application to finish running." This message is NOT an error, but merely informing the user that the computation has not yet finished. Because the actual computation for the analysis may be performed in R (such as the case for hierarchical clustering), the Excel application must wait until the computation has finished in R before Excel can continue on. When this message appears, the user can ignore the message and just keep waiting for the computation to finish. Even if the user does not click OK to dismiss the message, the message will disappear on its own. Or if the user does not want to wait, then the user may choose to stop the computation, by following the instructions given above in the section Stopping a computation after it has started running.

## Collating data using old collation dialogs

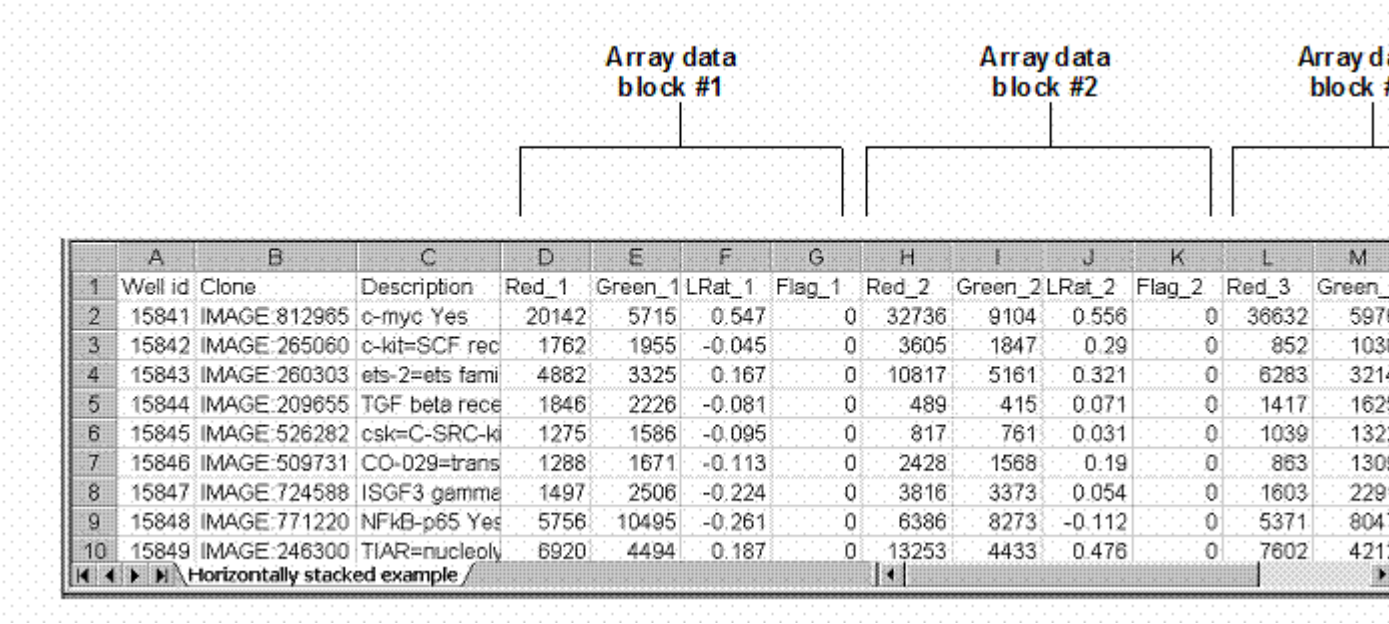
Prior to v3.1, BRB-ArrayTools used collation dialogs in which users must enter the format specifications directly, primarily by specifying column numbers where data elements were to be found. The collation dialogs used in previous versions are no longer necessary, since v3.1 now has a data import wizard which automatically reads the input files and displays the columns which are to be specified. However, the old collation dialogs are still supplied with BRB-ArrayTools v3.1 for backward-compatibility.

### Example 1 - Experiments are horizontally aligned in one file

**Format of the raw data files**

In this file format, the expression data from each array is represented by a block of columns in the data file, and the array data blocks are horizontally aligned in one file. There may be other columns, such as gene identifiers, that precede the array data blocks. However, there should be no other miscellaneous data columns following the last array data block. If there are any other data columns following the last array data block, then the user should either delete those last columns or move them to the beginning of the file so that they precede the first array data block.

The file may be a tab-delimited ASCII text file, or an Excel spreadsheet (a single worksheet within an Excel workbook). If the file is an Excel spreadsheet, then BRB-ArrayTools will automatically convert it to text format. The following figure shows an example of a horizontally aligned expression data file with four columns in each data block:




| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Well id | Clone | Description | Red_1 | Green_1 | LRat_1 | Flag_1 | Red_2 | Green_2 | LRat_2 | Flag_2 | Red_3 | Green_ |
| 2 | 15841 | IMAGE:812965 | c-myc Yes | 20142 | 5715 | 0.547 | 0 | 32736 | 9104 | 0.556 | 0 | 36632 | 597 |
| 3 | 15842 | IMAGE:265060 | c-kit=SCF rec | 1762 | 1955 | -0.045 | 0 | 3605 | 1847 | 0.29 | 0 | 852 | 103 |
| 4 | 15843 | IMAGE:260303 | ets-2=ets fami | 4882 | 3325 | 0.167 | 0 | 10817 | 5161 | 0.321 | 0 | 6283 | 321 |
| 5 | 15844 | IMAGE:209655 | TGF beta rece | 1846 | 2226 | -0.081 | 0 | 489 | 415 | 0.071 | 0 | 1417 | 162 |
| 6 | 15845 | IMAGE:526282 | csk=C-SRC-ki | 1275 | 1586 | -0.095 | 0 | 817 | 761 | 0.031 | 0 | 1039 | 132 |
| 7 | 15846 | IMAGE:509731 | CO-029=trans | 1288 | 1671 | -0.113 | 0 | 2428 | 1568 | 0.19 | 0 | 863 | 130 |
| 8 | 15847 | IMAGE:724588 | ISGF3 gamma | 1497 | 2506 | -0.224 | 0 | 3816 | 3373 | 0.054 | 0 | 1603 | 229 |
| 9 | 15848 | IMAGE:771220 | NFkB-p65 Yes | 5756 | 10495 | -0.261 | 0 | 6386 | 8273 | -0.112 | 0 | 5371 | 804 |
| 10 | 15849 | IMAGE:246300 | TIAR=nucleoly | 6920 | 4494 | 0.187 | 0 | 13253 | 4433 | 0.476 | 0 | 7602 | 421 |

The following figure shows the experiment descriptor file. The experiment descriptor file describes the samples used in the dataset. This particular dataset is a replication study. Each row of the experiment descriptor file describes the sample on one array. The order of the experiments listed in the experiment descriptor file are assumed to correspond to the order of the array data blocks in the expression data file. Thus, array data block #1 will be matched with HSOC2px-10 in row 2 of the experiment descriptor file, array data block #2 will be matched with HSOC2px-11 in row 3 of the experiment descriptor file, etc. The Reverse descriptor in column D indicates the experiments on which the fluors were reversed.

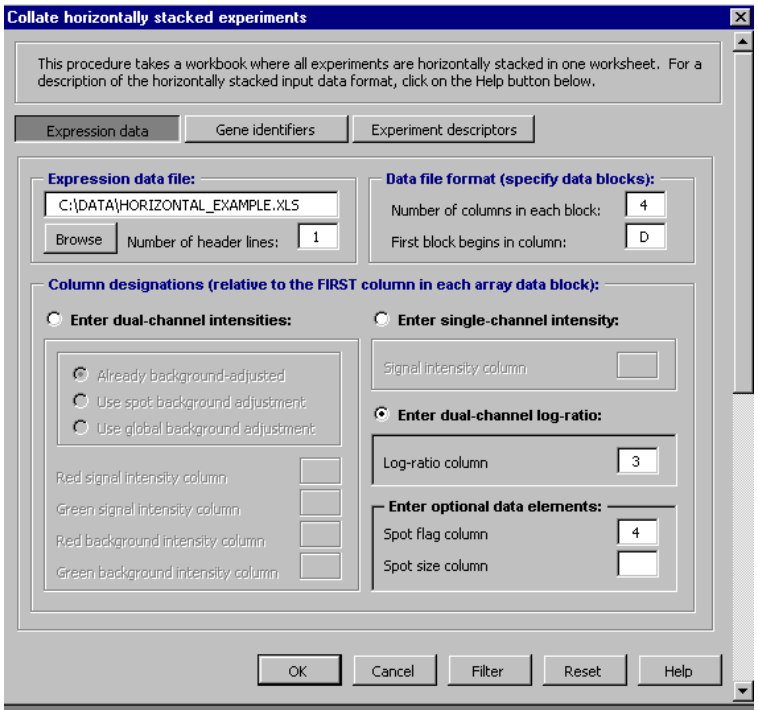
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Name | Red Probe | Green Probe | Reverse | |
| 2 | HSOC2px-10 | Sample X | Control | 0 | |
| 3 | HSOC2px-11 | Sample X | Control | 0 | |
| 4 | HSOC2px-12 | Sample X | Control | 0 | |
| 5 | HSOC2px-14 | Control | Sample x | 1 | |
| 6 | HSOC2px-15 | Control | Sample x | 1 | |
| 7 | | | | | |

Experiment descriptors

**Collating parameters**

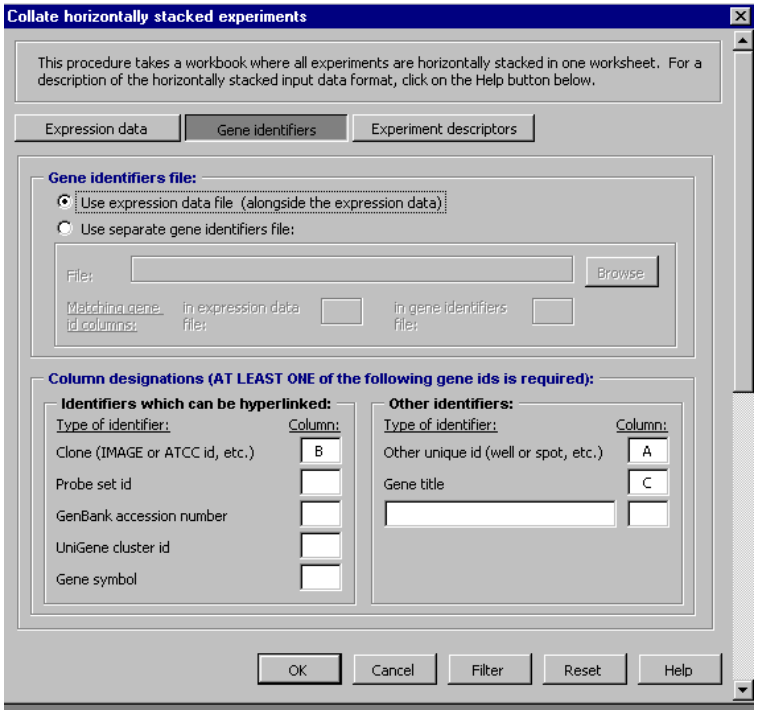
There are two ways in which the data may be specified in the collating dialog form. One way is to enter the red and green signal columns, and let BRB-ArrayTools compute the log-ratio. Another way is to enter the log-ratio column directly. The user is not permitted to enter both the log-ratio and the dual-channel signals, since the log-ratios can be computed once the dual-channel signals are entered. It is preferred that the dual-channel signals be entered (if they are available) rather than the log-ratio, since BRB-ArrayTools can use the signal intensities for filtering, lowess (intensity-based) normalization, and creating diagnostic scatterplots. If the log-ratios are entered directly, then these functionalities are not available to the user.

The figure below shows the collating parameters which would be entered on the **Expression data** page of the collating dialog box to describe the horizontally aligned data file shown above, if the signal intensities were to be entered rather than the log-ratio. The user should browse for or type in the name and path of the file that contains the expression data. Note that the column designations of the red, green, and spot flag variables are *relative to the first column of the array data block*. In other words, a designation of "4" for the spot flag column means that the spot flag is in the *fourth* column of each array data block. Any extra columns within the array data blocks that are not specifically designated in the **Data variables** section of the collating dialog box, such as the third column (the log-ratios), will be ignored during the collating step. Since BRB-ArrayTools automatically calculates the log-ratio from the red and green signals, it is not necessary to enter the log-ratios as input.

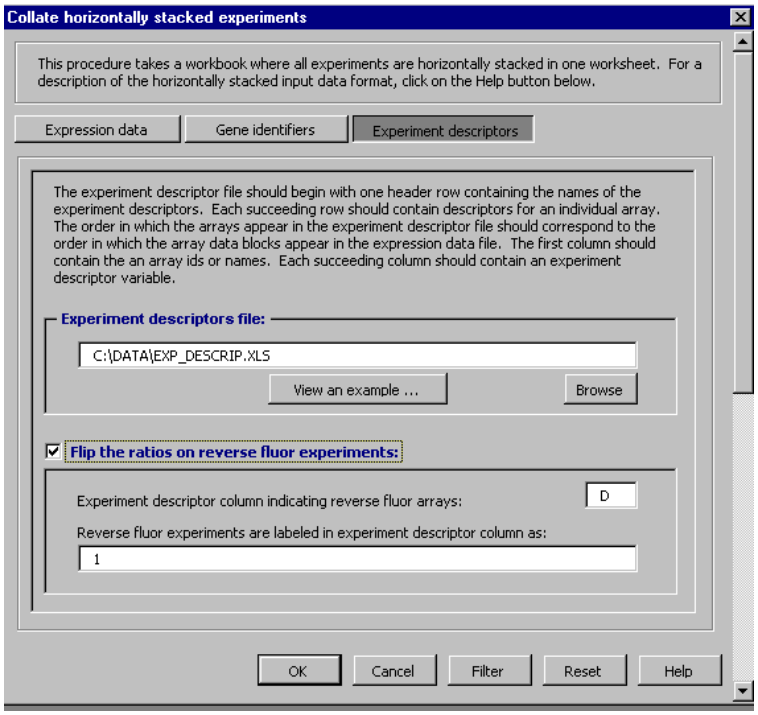



The figure below shows the collating parameters which would be entered on the **Expression data** page of the collating dialog box to describe the horizontally aligned data file shown above, if the log-ratios were to be entered directly instead of the signal intensities.

The figure below shows the collating parameters which would be entered on the **Gene identifiers** page of the collating dialog box. In this case, the gene identifiers are found in the same rows alongside the expression data, not in a separate file. The gene identifiers are specified by their column designation under the **Column format** section, which are given relative to the file specified in the **File** section.



The figure below shows the collating parameters which would be entered on the **Experiment descriptors** page of the collating dialog box. The user should browse for or type in the file and path name of the experiment descriptor file. In this case, any array with a label of `1` in column **D** of the experiment descriptor file would have its intensity ratios "flipped" so that the log-ratios in that array are "reversed" to have the opposite sign.




The **Filter** dialog box, which allows the user to change the default filtering parameters, will be reviewed in a separate section of this manual. After all the collating and filtering parameters have been entered, the user will be prompted to enter a name for the **project folder** to be used exclusively for this dataset, and to enter a name for the **collated project workbook** to be created.

## Example 2 - Experiments are in separate files

### Format of the raw data files

In this file format, the expression data from each array is stored in a separate file. All files must have the same column format, though the genes contained in the rows of the separate files may differ. Each file may be a tab-delimited ASCII text file, or an Excel spreadsheet (a single worksheet within an Excel workbook). If the files are a Excel spreadsheets, then BRB-ArrayTools will automatically convert them to text format. The `BreastSamples` dataset enclosed with this software package is an example of this data format. This dataset was obtained from published data by Ross, et al.

To view the `BreastSamples` dataset, unpack the `BreastSamples.zip` file using WinZip or other utility. When you unpack the dataset, you should get a folder called `BreastSamples`, containing a gene identifier file called `GeneIds.xls`, an experiment descriptor file called `ExpDesc.xls`, and a subfolder called `ExpressionData`. Inside the `ExpressionData` subfolder, there are thirteen expression data files. The expression data files in this dataset are already aligned.

The following figure shows the format of each of the expression data files:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | CH1D | CH2DN | FLAG | SPOT | |
| 2 | 787 | 915 | 0 | 49 | |
| 3 | 5642 | 6874 | 0 | 50 | |
| 4 | -3 | 25 | 0 | 51 | |
| 5 | 359 | 423 | 1 | 52 | |
| 6 | 15497 | 13586 | 0 | 53 | |
| 7 | 111 | 21 | 0 | 54 | |
| 8 | 16 | 63 | 0 | 55 | |
| 9 | 12578 | 5406 | 0 | 56 | |
| 10 | 2012 | 2029 | 0 | 57 | |
| 11 | 371 | 298 | 0 | 58 | |
| 12 | 57 | 24 | 0 | 59 | |

The following figure shows the format of the gene identifiers file:

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | Spot | Clone | Description | GB acc | | |
| 2 | 49 | 60204 | Homo sapiens C2H2 zinc finger protein pseudogene, mRNA sequence | T39154, T40438 | | |
| 3 | 50 | 60436 | RPL3 Ribosomal protein L3 Chr.22 | T39295, T40510 | | |
| 4 | 51 | 60218 | EST | T39165, T40450 | | |
| 5 | 52 | 60209 | EST | T39163, T40448 | | |
| 6 | 53 | 60664 | EST | T39448, T40595 | | |
| 7 | 54 | 60932 | CSH1 Chorionic somatomammotropin hormone 1 (placental lactogen) Chr.17 | T39603, T40692 | | |
| 8 | 55 | 60955 | EST | T39626, T40701 | | |
| 9 | 56 | 61539 | Cytochrome P450, subfamily IIC (mephenytoin 4-hydroxylase), polypeptide 9 | T40065, T40987 | | |

GeneIds

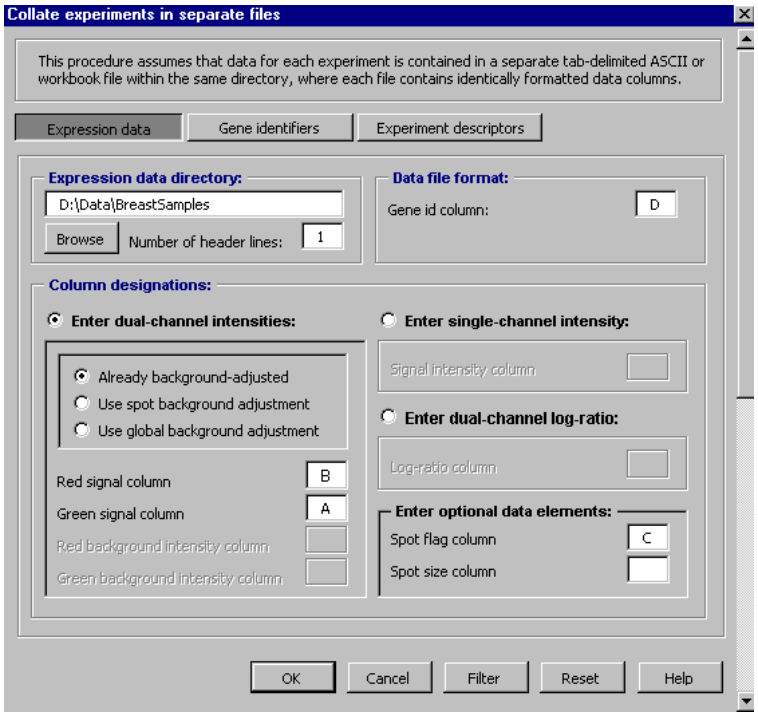
The following figure shows the format of the experiment descriptors file:

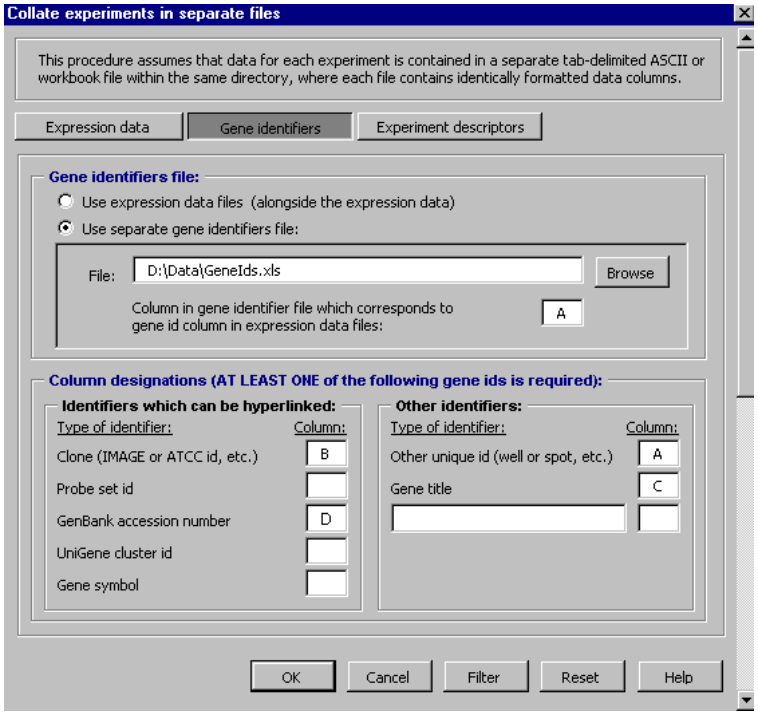
| | A | B | C |
|---|---|---|---|
| 1 | ExpId | Type | |
| 2 | BC16 | Tissue | |
| 3 | BC2 | Tissue | |
| 4 | BC2_Lymph_Node | Tissue | |
| 5 | BT-549_CL5013_BREAST | Cell-line | |
| 6 | HS_578T_CL5006_BREAST | Cell-line | |
| 7 | MCF7_CL5001_BREAST | Cell-line | |
| 8 | MCF7A-repro | Cell-line | |
| 9 | MCF7D-repro | Cell-line | |
| 10 | MDA-MB-231_CL5005_BREAST | Cell-line | |
| 11 | MDA-MB-435_CL5011_BREAST | Cell-line | |
| 12 | MDA-N_CL5012_BREAST | Cell-line | |
| 13 | Normal_Breast | Tissue | |
| 14 | T-47D_CL5014_BREAST | Cell-line | |
| 15 | | | |

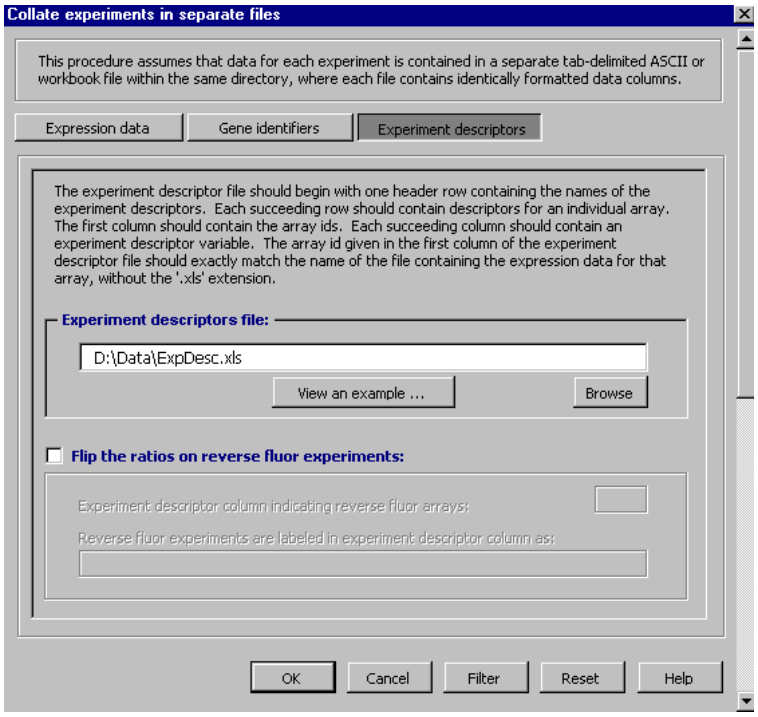
Experiment descriptors

## Collating parameters

The following three figures show the collating parameters that would be entered on the **Expression data** page, **Gene identifiers** page, and **Experiment descriptors** page of the collating dialog box, for collating the `BreastSamples` data using `SPOT` as the unique gene identifier that links the expression data with the gene identifiers.






The **Filter** dialog box, which allows the user to change the default filtering parameters, will be reviewed in a separate section of this manual. After all the collating and filtering parameters have been entered, the user will be prompted to enter a name for the **project folder** to be used exclusively for this dataset, and to enter a name for the **collated project workbook** to be created.

## Reporting bugs

To make comments or ask questions of a general nature, please post a message on the message board at the BRB-ArrayTools website:

http://linus.nci.nih.gov/BRB-ArrayTools.html

Bug reports should be e-mailed to the BRB-ArrayTools Development Team at: arraytools@emmes.com. Because many emails get automatically deleted as spam, please follow these guidelines to ensure that your email gets past the filter:

1) Always include an appropriate subject header which indicates the topic of the email.
2) If you are sending attachments, then please send them in a SEPARATE email from the main text, so that we will know in case the email with attachments gets deleted by our virus filter.

Due to limited resources, we cannot guarantee that we will be able to diagnose and fix all bugs reported by users.  However we do try our best to respond to users within a reasonable time, and to advise the user or fix the problem whenever we can.

In v3.5, we have implemented a log file called “DataParams” that stores the parameter options used at the time of importing the data and running various analysis tools. This file is stored in the “Binary” folder under the project folder. When reporting bugs to the BRB-ArrayTools Development Team, it would be helpful to send this “DataParams” file as an attachment to the bug report .

# References

## Published references

S Dudoit, J Fridlyand, TP Speed; Comparison of discrimination methods for the classification of tumors using gene expression data, Journal of the American Statistical Association 97:77-87, 2002

I Hedenfalk., D Duggan, Y Chen, M Radmacher, M Bittner, R Simon, P Meltzer, B Gusterson, M Esteller, M Raffeld, et al. Gene expression profiles of hereditary breast cancer, New England Journal of Medicine 344:539-548, 2001.

EL Korn, JF Troendle, LM McShane and RM Simon. Controlling the number of false discoveries: Applications to high-dimensional genomic data. Journal of Statistical Planning and Inference (In Press).

LM McShane, MD Radmacher, B Freidlin, R Yu, MC Li and RM Simon. Methods of assessing reproducibility of clustering patterns observed in analyses of microarray data. Bioinformatics 18:1462-1469, 2002.

P Pavlidis, J Qin, V Arango, JJ Mann and E Sibille. Using the Gene Ontology for microarray data mining: A comparison of methods and application to age effects in human prefrontal cortex. Neurochemical Research 29:1213-22, 2004.

MD Radmacher, LM McShane and RM Simon. A paradigm for class prediction using gene expression profiles. Journal of Computational Biology, 9:505-511, 2002.

D Ross, U Scherf, M Eisen, C Perou, C Rees, P Spellman, V Iyer, S Jeffrey, M Van de Rijh, M Waltham et al. Systematic variation in gene expression patterns in human cancer cell lines, Nature Genetics 24:227-235, 2000.

RM Simon, EL Korn, LM McShane, MD Radmacher, GW Wright and Y Zhao. Design and Analysis of DNA Microarray Investigations, Springer, New York NY, 2003.

V Tusher, R Tibshirani and G Chu. Significance analysis of microarrays applied to transcriptional responses to ionizing radiation. Proceedings of the National Academy of Sciences USA 98:5116-5121, 2001.

GW Wright and RM Simon. A random variance model for detection of differential gene expression in small microarray experiments. Bioinformatics 19:2448-2455, 2003.

## Technical reports

In addition, the following two references may be found as technical reports from the BRB website: http://linus.nci.nih.gov/~brb/TechReport.htm

Technical Report 001:A paradigm for class prediction using gene expression profiles. by Michael D. Radmacher, Lisa M. Mcshane, and Richard Simon

Technical Report 002:Methods for assessing reproducibility of clustering patterns observed in analyses of microarray data. by Lisa M. McShane, Michael D. Radmacher, Boris Freidlin, Ren Yu, Ming-Chung Li, and Richard Simon

# Acknowledgements

The BRB-ArrayTools Development Team is led by Dr. Richard Simon and current team members are Dr. Ming-Chung Li, Dr. Lori Long, Dr. Qian Xie, Robin Kramer and Qihao Qi. Past developers who have contributed to the software are Supriya Menezes, Michael Ngan, Amy Peng Lam, Dr.Yong Li, Dr. Leonid Gibansky, Deyun Pan and Paul Shrabstein. This software uses many algorithms and programs developed by various members of and people associated with the Biometrics Research Branch. Many code fragments and hints were also taken from the Excel Developer Tip Archives (http://www.j-walk.com/ss/excel/tips/index.htm) from JWalk & Associates, Inc.

Acknowledgements also go to the R Core Development Team for their production of the R software (http://cran.r-project.org). For details, please see the 'Readme.doc' file. Support Vector Machine algorithm implemented in the BRB ArrayTools was developed by Chih-Chung Chang and Chih-Jen Lin (see http://www.csie.ntu.edu.tw/~cjlin/libsvm/).

Cluster 3.0 software was developed by Hoon et. al (M. J. L. de Hoon, S. Imoto, J. Nolan, and S. Miyano: Open Source Clustering Software. *Bioinformatics*, 2003, in press.). It is represents an enhanced version of Cluster, which was originally developed by Michael Eisen of Berkeley Lab.

To view the clustering results generated by Cluster 3.0, we use Alok Saldanha's Java TreeView. Java TreeView is not part of the Open Source Clustering Software.

# License

Cluster 3.0 is covered by the original Cluster/TreeView license.

Support Vector Machine algorithm implemented in the BRB ArrayTools was developed by Chih-Chung Chang and Chih-Jen Lin (see http://www.csie.ntu.edu.tw/~cjlin/libsvm/ ). Redistribution and use in source and binary forms, with or without modification, is

permitted as described in the Libsvm_copyright.doc file distributed together with the BRB ArrayTools.